CONTRACT NAS9-9953 MSC 02471
DRL NO: MSC-T-575, LINE ITEM 72

SD 72-SA-0114-5

MODULAR space station

PHASE B EXTENSION

INFORMATION MANAGEMENT ADVANCED DEVELOPMENT FINAL REPORT

Volume V: Software Assembly

PREPARED BY PROGRAM ENGINEERING
JULY 31, 1972

Space Division
North American Rockwell

12214 Lakewood Boulevard, Downey, California 90241

CONTRACT NAS9-9953 MSC 02471
DRL NO: MSC-T-575, LINE ITEM 72

SD 72-SA-0114-5

MODULAR
space station
PHASE B EXTENSION

INFORMATION MANAGEMENT ADVANCED DEVELOPMENT FINAL REPORT

Volume V: Software Assembly

31 JULY 1972
PREPARED BY PROGRAM ENGINEERING

Approved by

James Madewell
Director
Space Applications Programs

Space Division
North American Rockwell

12214 Lakewood Boulevard, Downey, California 90241

TECHNICAL REPORT INDEX/ABSTRACT

CESSION NUMBER						DOCUMENT SECURITY CLASSIFICATION	
						UNCLASSIFIED	

ITLE OF DOCUMENT	LIBRARY USE ONLY
NFORMATION MANAGEMENT ADVANCED DEVELOPMENT FINAL REPORT, olume V, Software Assembly	
UTHOR(S)	
erber, C. R., et. al.	

ODE	ORIGINATING AGENCY AND OTHER SOURCES	DOCUMENT NUMBER
NO85282	Space Division of North American Rockwell Corporation, Downey, California	SD 72-SA-0114-5

UBLICATION DATE	CONTRACT NUMBER
July 31, 1972	NAS9-9953

ESCRIPTIVE TERMS

MODULAR SPACE STATION, *INFORMATION MANAGEMENT, ADVANCED DEVELOPMENT, *SOFTWARE, *DATA STORAGE, HIGHER ORDER LANGUAGE, *COMPUTER PROGRAM, *STANDARDS, CONVENTIONS, *SPECIFICATIONS

BSTRACT

HIS DOCUMENT IS VOLUME V OF THE FINAL REPORT OF THE MODULAR SPACE TATION ADVANCED DEVELOPMENT STUDY. IT SUMMARIZES THE RESULTS OF ASKS WHICH DEVELOPED UNIFORM COMPUTER PROGRAM STANDARDS AND ONVENTIONS FOR THE SPACE STATION PROGRAM, DEVELOPED A COMPUTER ROGRAM SPECIFICATION HIERARCHY, DEFINED A COMPUTER PROGRAM EVELOPMENT PLAN, MADE RECOMMENDATIONS FOR THE EFFECTIVE UTILIZATION F ALL OPERATING ON-BOARD SPACE STATION PROGRAM RELATED DATA ROCESSING FACILITIES, AND DEVELOPED A PRELIMINARY COMPUTER PROGRAM PECIFICATION FOR THE MSS CENTRAL PROCESSOR EXECUTIVE PROGRAM.

FOREWORD

This document is one of a series required by Contract NAS9-9953, Exhibit C, Statement of Work for the Phase B Extension-Modular Space Station Program Definition. It has been prepared by the Space Division, North American Rockwell Corporation, and is submitted to the National Aeronautics and Space Administration's Manned Spacecraft Center, Houston, Texas, in accordance with the requirements of the Data Requirements List (DRL) MSC-T-575, Line Item 72.

This document is Volume V of the Modular Space Station Information Management System Advanced Development Technology Report, which has been prepared in the following five volumes.

I	Summary	SD 72-SA-0114-1
II	Communications Terminal Breadboard	SD 72-SA-0114-2
III	Digital Data Bus Breadboard	SD 72-SA-0114-3
IV	Data Processing Assembly	SD 72-SA-0114-4
V	Software Assembly	SD 72-SA-0114-5

CONTENTS

ILLUSTRATIONS

TABLES

ABBREVIATIONS

ADS	Advanced Data System
ADT	Advanced Development Technology
ADTX	Advanced Development Technology Extension
AFC	Automatic Frequency Control
AGC	Automatic Gain Control
AN	Autonetics (Division of North American Rockwell Corporation)
BPF	Bandpass Filter
bps	Bits Per Second
CAIRS	Computer-Assisted Interactive Resource Scheduling
CCIR	International Radio Consultative Committee
CDR	Critical Design Review
CEI	Contract End Item
CLASP	Computer Language for Aeronautics and Space Programming
CM	Command Module (Apollo)
COMPOOL	Common Pool (of Data)
CP	Central Processor or Circular Polarization
C.P.	Computer Program
CPCEI	Computer Program Contract End Item
CPCI	Computer Program Configuration Item
CPDF	Computer Program Development Facility
CPIC (A)	Computer Program Integration Contractor (Agency)
CPT&E	Computer Programming Test and Evaluation
CR	Change Report
CRT	Cathode-Ray Tube (Display)
CSS	Crew Subsystem
CTB	Communications Terminal Breadboard
CTF	Central Test Facility
DACS	Data Acquisition and Control Subassembly
dB	Decibel
DBCU	Data Bus Control Unit
dBm	Decibel Referred to One Milli-Watt
dBW	Decibel Referred to One Watt
DCR	Design Change Request
DDB	Digital Data Bus
Demux	De-Multiplex(er)
DMS	Data Management System
DPA	Data Processing Assembly
DPSK	Dual Phase Shift Keying
DRSS	Data Relay Satellite System

ECP	Engineering Change Proposal
EDF	Experiment Data Facility
EEM	Engineering Evaluation Model
EEMP	Engineering Evaluation Model Processor
EIRP	Effective Isotropic Radiated Power
EMC	Electromagnetic Compatability
EMI	Electromagnetic Interference
EOS	Earth Orbital Shuttle
EOSS	Earth Orbital Space Station
EPS	Electrical Power Subsystem
ETC/LSS or ECLSS	Environment Control and Life Support Subsystem
EVA	Extra-Vehicular Activity
EXT	External
E_b/N_o	Energy Per Bit to Noise Density Ratio
FACS	Facsimile
FDM	Frequency-Division Multiplex
FM	Frequency Modulation
FQT	Formal Qualification Test
G&CS	Guidance and Control Subsystem
GFE	Government Furnished Equipment
GHz	Giga-Hertz
GOA	Gated Operational Amplifier
HAL	Higher-Order Aerospace Programming Language
HOL	Higher-Order Language
HOLM	Higher-Order Language Machine
Hz	Hertz
IF	Intermediate Frequency
IFRU	In-Flight Replaceable Unit
IM	Intermodulation Products
IMS	Information Management System
IMSIM	Information Management Simulation
IOC	Initial Operational Capability
IOCB	Input-Output Control Block
IOU	Input-Output Unit
I/O	Input-Output
IPA	Intermediate Power Amplifier
IQL	Interactive Query Language
IR	Infra-Red
ISS or IMS/S	Information (Management) Subsystem
ITT	International Telephone and Telegraph
K-words	Thousands of (Computer) Words
K-EAPS	Thousands of Equivalent-Add Operations Per Second
K-bps	Thousands of Bits Per Second
KHz	Kilohertz

LEM	Lunar Excursion Module
LM	Lunar Module
LNA	Low Noise Amplifier
LO	Local Oscillator
LPF	Low Pass Filter
M1, M2	(Computer) Memory Designation
Mbps	Megabits Per Second
MCB	Module Control Block
MHz	Megahertz
MOF	Mission Operations Facility
MOL	Manned Orbiting Laboratory
MSC	Manned Spacecraft lenter
MSFN	Manned Space Flight Network
MSS	Modular Space Station
MUX	Multiplexer
mW	Milli-Watts
MW	Microwave
mV	Milli-Volts
NF	Noise Figure
OBCO	On-Board Checkout
OCC	Operations Control Center (On-Board)
ODM	Operational Data Management
OM	Operating Memory
PA	Power Amplifier
PCM	Pulse Code Modulation
PDR	Preliminary Design Review
PL/1	Procedure Language
PM	Phase Modulation
PN (PRN)	Pseudo Random Noise
ppm	Parts Per Million
POT	Preliminary Qualification Tests
PSK	Phase Shift Keying
RAM	Research and Applications Module
RACU	Remote Acquisition and Control Unit
RCS	Reaction Control Subsystem
RF	Radio Frequency
RHCP	Right-Hand Circular Polarization
RPU	Remote Processing Unit
Rx	Receive
S&C	Standards and Conventions
SCCB	Software Configuration Control Board
SCN	Specification Change Notice
SD	Space Division (of North American Rockwell Corporation)
SDC	Systems Development Corporation

S/N	Signal to Noise Ratio
SOW	Statement of Work
SPL	Space Programming Language
SRD	Step-Recovery Diode
SSCB	Solid-State Circuit-Breaker
SSS	Structures Subsystem
STE	Support Test Equipment
TAV	Test and Validation (Programs)
TBD	To Be Determined
TCXO	Temperature-Controlled Crystal Oscillator
TDA	Tunnel Diode Amplifier
TDM	Time Division Multiplexing
TDRS	Tracking and Data Relay Satellite
TIP	Test and Integration Plan
TLM, TM	Telemetry
TOOL	Test Operations Oriented Language
TRW	Thompson Ramo Woolridge Corporation
TT&C	Telemetry, Tracking and Control
TWT	Traveling Wave Tube
TWTA	Traveling Wave Tube Amplifier
Tx	Transmit
USB (E)	Unified S-Band (Equipment)
UV	Ultra-Violet
VDD	Version Description Document
VHF	Very High Frequency
VSB	Vestigal Side Band
VSWR	Voltage Standing Wave Ratio

LIST OF INTERIM REPORTS

AA-101	DPA Flow Diagrams, September 1971
AA-102	DPA Throughput and Authority Analysis, February 1972
AA-103	DPA Configuration Selection, April 1972
AS-101	Modular Space Station Computer Program Standards and Conventions, December 1971
AS-102	Modular Space Station Computer Program Specification Tree, February 1972
AS-103	Modular Space Station Computer Program Development, Test and Configuration Control Plan, May 1972
AS-104	Modular Space Station Computer-Assisted Resource Allocations and Utilization Recommendations, June 1972
CTB-101	Concepts for Multiple RF Link Mechanization, May 1971
CTB-103	Antenna-Mounted Electronics Component Design, October 1971
CTB-105/106	CTB Integration and Test and Operations Manual, June 1972
DB-101	Parametric Data for Bus Design, May 1971
DB-103	Component Performance Requirements, Schematics and Layout Drawings, December 1971
DB-104	Digital Data Bus Breadboard Final Report, May 1972
DD-102	Modular Space Station Data Processing Assembly Parametric Evaluation of Subsystems Input/Output Interface, June 1971

DD-103	Modular Space Station Data Acquisition and Control Subassembly Model Configuration (SD 71-233), July 1971
DP-101	Data Processing Assembly Configuration (Preliminary), June 1971
DP-102	Data Processing Assembly Supervisor Specification, May 1972
DP-103	DPA Processor Performance Requirements (Preliminary), August 1971
DP-103	DPA Processor Final Description, May 1972
DP-104	EEM DMS Processor Development Plan, June 1972
DP-105	Data Acquisition and Control Redundancy Concepts, August 1971
DP-106	Application of Redundancy Concepts to DPA, January 1971
DP-107	Data Acquisition and Control Subassembly Breadboard Design Requirements, October 1971
DP-108	Data Bus Control Unit Performance Requirements, January 1972
DP-109	Data Bus Control Design Reports, March 1971
DP-110	DBCU Acceptance Report (to be published)
EL-277	Bulk Storage Development Plan
IB-101	DPA Internal Flow and Traffic Pattern, May 28, 1971
ICD #TRW 20549	Interface Control Document - Data Bus Modem/RACU, Revision A, January 17, 1972
ICD #AN 26465	Interface Control Document - Data Bus Controller Unit to Buffer I/O, Revision January 21, 1972
MD-101	Mass Memory Parametric Data
RF-101	Modular Space Station Communications Terminal Breadboard Preliminary System Specification, October 1971

SA-101	Central Processor Operational Analysis, September 30, 1971
SA-102	Central Processor Memory Organization and Internal Bus Design, December 30, 1971
SD 71-227	Automatic Control and Onboard Checkout Final Study Report

ACKNOWLEDGEMENTS

The following persons have participated in the conduct of the IMS ADT tasks, and have contributed to this report:

C. W. Roberts	Experiment/Electronics Manager
C. R. Gerber	Information Systems Project Engineer
B. A. Logan, Jr.	Information Systems
E. Mehrbach	Information Systems
D. W. Brewer	Information Systems
V. R. Hodgson	Information Systems

The following subcontractors have supported the IMS ADT tasks in specialized areas:

International Tel. & Tel. Nutley, New Jersey B. Cooper, Proj. Mgr.	Communications Terminal B.B. Data Bus BB
Intermetrics Cambridge, Mass. J. Miller, Prog. Mgr.	Data Processing Assy
System Develop. Corp. Santa Monica, Calif. R. Bilek, Prog. Mgr.	Data Processing Assy Software Assy
Gen'l Electric Corp. Valley Forge, Pa. R. Kirby, Prog. Mgr.	Bulk Storage Technology
NR-Autonetics Anaheim, Calif. J. Jurison, Proj. Mgr.	Data Bus BB Data Processing Assy

1.0 INTRODUCTION

1.0 INTRODUCTION

This volume presents the results of several inter-related studies conducted during the MSS Phase B Preliminary Definition Study to carry the analysis and definition of the MSS Information Subsystem Software Assembly beyond the depth normally expected in a Phase B. These added studies were incorporated as an added special emphasis task (Task 4.2 of the SOW) designated as "Information Management System Advanced Development Technology". Since these added studies extended beyond the MSS Phase B Preliminary Definition itself, they are herein reported in five volumes, corresponding, roughly, to the MSS Information Subsystem Assemblies:

Volume I	Summary
Volume II	RF Communications Terminal BB
Volume III	Data Acquisition/Control Subassy BB
Volume IV	Data Processing Assembly
Volume V	Software Assembly

Volumes III, IV, and V are closely interrelated in that, together, they define the onboard Data Management functions; the reader is advised to examine all three volumes for completeness. In particular, the DPA Supervisory Specification (Volume V) and the DPA Central Processor Specification (Volume IV) have been closely coordinated.

The Modular Space Station incorporates a very large digital computer complex to assist the crew in accomplishing their assigned tasks. It was recognized from inception of the Phase B study that the definition of the design requirements for the computer complex would be sensitive to the software (computer programs) that was to run in it. To emphasize the importance of the software in the MSS program, it was defined to be an assembly, on the same level, but separate from the hardware assembly. It was also recognized that the assignment of functions to the several subassemblies (both hardware and software) would involve many tradeoffs to obtain a realistic configuration, and that it would be a mistake to define the hardware requirements without consideration of the software requirements.

Therefore, the MSS software assembly was included within the Advanced Development Task as a series of interrelated studies in concert with similar Data Processing Assembly (Volume IV) studies; these tasks were developed and conducted by representatives from SDC, AN, and SD as a cooperative effort. Figure 1-1 represents the study logic flow, and indicates which agency was assigned responsibility for providing the subtask documentation. Those subtasks that are distinctly software (94012-1 through 4 and 85170-2) were influenced by subtasks within the DPA (hardware) flow, as shown by the arrows.

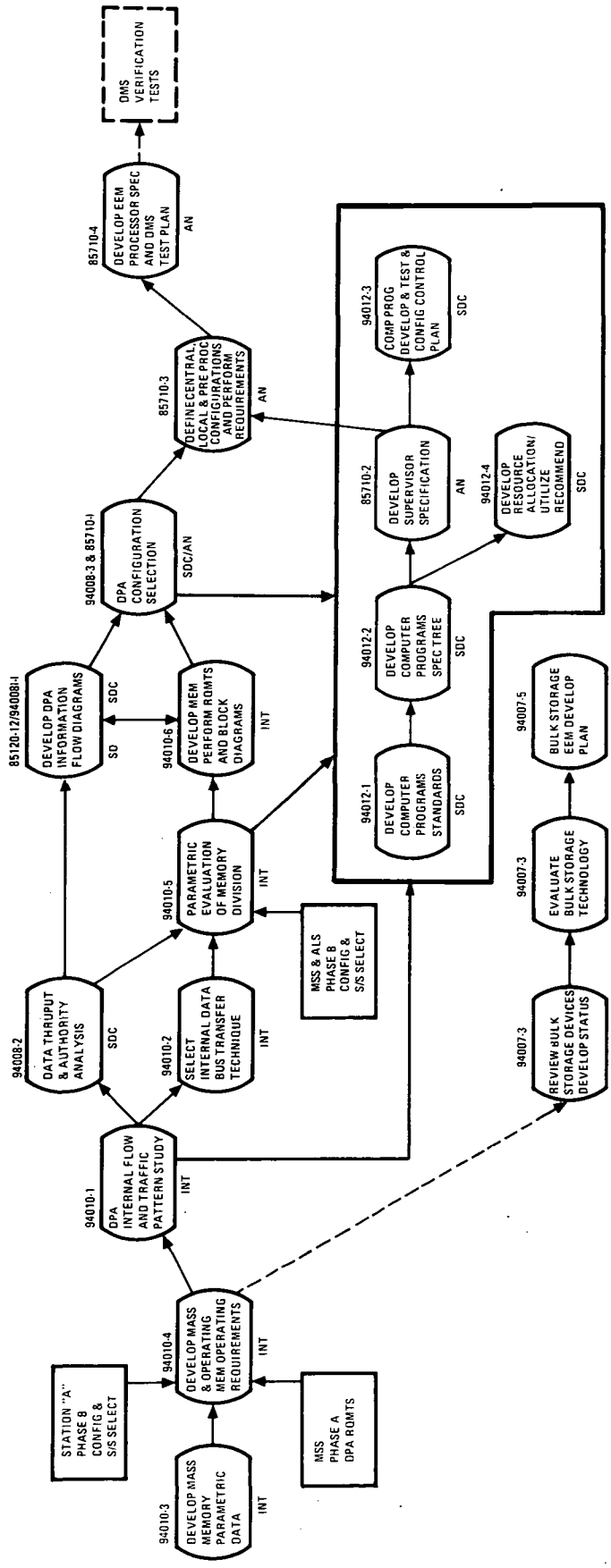


Figure 1-1. Information Management Advanced Development Data Processor and Software Study Plan

The intent of these tasks was to develop uniform computer program standards and conventions for the space station program, develop a computer program specification hierarchy, define a computer program development plan, make recommendations for the effective utilization of all operating onboard space station program related data processing facilities, and develop a preliminary computer program specification for the MSS CP executive program.

1. Develop Computer Program Standards. This task reviewed present computer program languages, assemblies, compilers, coding and data base conventions, specifications, validation, and documentation methodology, and presents recommendations for the MSS Computer Programs Standards and Conventions.

2. Develop Computer Program Specification Tree. This task defined the computer program system specification hierarchy (tree). In addition, a preliminary top level computer program system specification was prepared to identify all computer program documents and tapes in accordance with the standards recommended in Task 1.

3. Computer Program Development, Test, and Configuration Control Plan. This task provides a plan for development, validation and documentation of all computer programs. The plan includes recommendations for computer program configuration control, validation/acceptance criteria, and test bed facilities.

4. Resource Allocation and Utilization Recommendations. This task develops approaches for effective utilization of all MSS-related data processing facilities. Consideration was given to on-board data reduction/evaluation, and/or shuttle transfer-to-ground for ground-based data reduction, analysis and distribution. Computer-assisted methods and procedures for allocating resources (such as crew skills, consumables, sensors, etc.) within identifiable constraints, scheduling resupply and crew rotation, and long-range planning such as mission management, and short-range planning such as crew duty schedules, are defined.

5. Supervisory Specification. This task develops preliminary performance characteristics of computer program modules needed to control the central processor selected as part of the DPA configuration. Characteristics such as CP start-up, initialization, synchronization, "job" scheduling, input/output control, reconfigurability, CP monitoring and self test, and CP shutdown/restart were considered. The objective was to develop an optimum supervisory (executive) design from both hardware and program considerations for the modular space station (MSS) application.

These five subtasks are by no means exhaustive of the necessary preliminary studies of the MSS Software Assembly that need to be completed before any actual computer programs are written; they are, perhaps, the very tip of the iceberg that is software cost avoidance. Historically, the costs of developing software for extensive computer systems have been "buried" under other titles, and seldom indicate that the man-hours and other cost dollars attributable to hardware (the computer system) are usually exceeded by the costs for the programs

that are to be run in the computer. Recognition of this situation caused NR and MSC to identify all the MSS Software as a separate assembly of the on-board Information Management System so as to further identify and devise means to avoid excessive cost buildup. The result of this choice has been to highlight the tendency of most projects to accelerate the specification and development of the hardware, and avoid the impact this has upon the software.

Identification of the components of software cost buildup strongly influenced the Preliminary Design of the MSS Data Processing Assembly. Typically, the factors of limited computer speed and memory capacity were listed by several cognizant subcontractors as a strong driver on software costs; the MSS DPA is, therefore, "over sized" in terms of identifiable Station Operations needs (see Volume IV). Another factor listed by many was the lack of experienced computer program designers; the MSS Preliminary Design was based upon the assumption that if the programs (routines) could be written in a Higher-Order Language, in sufficiently small "modules", this factor could be reduced considerably in influence. However, in order to accomplish this desirable objective, the total must be divided into very small work packages, and governed by standardized terminology and procedures, and planned to be effected during a progressive, unhurried time scale, and controlled with the same attention to quality and specificity as any of the hardware developments. It was to these points that the IMS ADT Software Assembly studies were addressed.

The body of this volume is made up of extracts from the several subtask reports. Each of the five sections will be preceded by the subtask description as originally defined, with an explanation of any modification or redirection that was needed, and followed by an evaluation and recommendation for its utilization in future efforts.

2.0 COMPUTER PROGRAM STANDARDS AND CONVENTIONS

2.0 COMPUTER PROGRAM STANDARDS AND CONVENTIONS

A review of present computer program languages, assemblies, compilers, coding and data base conventions, specification, validation, and documentation methodology was conducted to enumerate the standards applicable to the production and management of all modular space station computer programs. Typical of the classes of S&C defined are: language, language processors characteristics, data base, data cards, job control language, tape standards, documentation format and content, executive, messages, error detection, operation requests, call sequences, symbology, intercomputer interfaces, program modularization, and other classes as identified during the study. This was accomplished in five steps:

Review Application. The operational concepts of the primary application area, the space station, were reviewed in light of the standards and conventions that will be required to support the activity. Additionally, the concepts of the supporting areas (space shuttle, experiment modules, and ground complex) were investigated.

Review Existing S&C. Compile a set of documents describing S&C that have been developed both by SDC and other organizations for computer program development and operation activities. These documents include, as a minimum, S&C developed by: Mellonics for use in the Advanced Data System (ADS) supporting MOL, SDC for use by the Air Force in the Satellite Control Facility, Lockheed Missile and Space Company (LMSC) for use in ADS, SDC for AWACS, the ABMDA Standards Committee, and American National Standards Institute (ANSI). These documents were reviewed to identify those S&C which are directly applicable to the space station.

Establish S&C Scope. The scope identifies the level of detail to which the standards and conventions would be developed, the applications which must be covered by the S&C, the rules for preparing the S&C document, an organizational layout of the S&C document, and the basic S&C terminology to be used.

Recommended Method of Implementation. The management procedures necessary to control S&C after the initial publication of an approved manual are recommended. They cover updating and maintaining the manual and enforcement of the S&C.

Review Results and Prepare S&C Manual. The preliminary S&C document was reviewed by NR, MSC, and other subcontractors to solicit critiques and to coordinate the effort. The results were recorded in a preliminary Standards and Convention Manual. (Note: This task was redirected to delete a manual format.)

For all large systems, software or hardware, the establishment and implementation of standards and good operating practices, conventions, is an ongoing process which continues to be refined throughout the life of the system. In this respect, the material presented is not intended to be final or all inclusive. However, it is intended to identify those areas of profitable standardization and good practice which can serve as a base for uniform design, development, and implementation.

The information management concept and baseline description of the ISS, as described in Section 3.0 of the Space Station Program Phase B Program Operations Plan, SD 70-132, were assumed for this standards and conventions effort. Also assumed were the Computer Program Management Structure, computer program design and development approach, and the computer program management requirements, as specified in Section 4.0 of that document.

2.1 BASIC DEFINITIONS

As used in this document, a standard is a definite rule or procedure which has been established by authority. A convention is a customary or agreed upon rule or procedure. These standards and conventions will be of prime importance to computer programmers, system designers, and software system managers.

Software can be defined as the computer programs, the programming aids, and the documentation required to produce and describe computer programs. This includes not only the operational programs but also support programs such as compilers, loaders, simulators, data reduction, analysis, and documentation programs.

Software techniques are the methods used to produce computer programs. They include state-of-the-art knowledge and all tools which are and have been applied in computer programming activities.

Common and/or frequently used areas of software technique applications are termed information processing functions. These functions identify a group of actions and/or activities required to meet system software requirements. From a review of MSS operational and support elements, with possible software involvement, fourteen basic information processing functions were identified. These are:

- Program Production
- Program Organization
- Documentation
- Program Testing
- File Management
- Symbol Manipulation
- Calculations
- Decision Making
- Bookkeeping
- Timing
- Message Formatting
- Simulation
- Personnel Management

The standards and conventions presented in this document are identified and discussed within a framework of twelve topic areas. Six of these are information processing functions, where a variety of software techniques are applicable and six are specific software techniques. This structure was selected as the most optimum to cover the subject of software standards and conventions for the MSS program.

The twelve major topic areas are:

- Languages
- Compilers
- Program Production
- System Organization
- Formats
- Technology and Abbreviations
- Symbols
- Units of Measurement and Conversion
- Documentation
- Testing and Validation
- Maintenance
- Management

Many of the subjects discussed are interrelated but the considerations to be made in their standardization are discussed separately with references made to related subjects. Recommended standard and convention statements are underlined following the discussion of each major or minor topic. In many cases the recommendations are not specific, but indicate the general aspects that should be considered for the MSS program. For these areas additional studies are required or finalized system design parameters are necessary before specific recommendations are made.

Table 2-1 (P2-13) summarizes the suggested baseline standard and conventions. The intent of this section is to define an initial set of computer program standards and conventions that can serve as a beginning to the standards and conventions guidelines that will be required for the MSS program.

2.2 COMPUTER PROGRAMMING LANGUAGES

Computer programs are discussed in respect to the locations in which they will be operated. These locations are: (1) onboard the space station, (2) at ground support facilities, and (3) at software development facilities. On-board computer programs will be primarily application and supervisory types. Those used by ground support facilities will be a mix of supervisory, application, and support. At software development facilities computer programs will be primarily support programs used to develop application and supervisory programs.

2.2.1 Onboard Computer Program

2.2.1.1 Functional Characteristics

A review and analysis of the MSS onboard information management functions indicate that the majority of activities will require the following types of calculations and/or operations:

- Three-dimensional orientation and positioning
- Sensor I/O manipulation
- Command, control, and data distribution

- Display formulation and presentation
- File, library and data bank interrogation and updating
- Console command formulation, interpretation and monitoring

The majority of operational phases and experiments will be conducted with application computer programs which perform computations from sensor inputs and formulate these computations into commands or data which are sent to component, subsystem, and system effectors. Sensor inputs can be hardware transducers, onboard crew members or ground based communications.

Onboard supervisory computer programs will manage the data processing system by controlling, coordinating and scheduling the operations to be performed. However, because of the operational environment, the mission requirements load and the conceptual organization of the data processing system, a highly sophisticated supervisory computer program package will be required. System concepts which will dictate the degree of complexity and sophistication required are:

- Multiprocessing
- Software modularization
- Optimization of machine load
- Interactive control
- Self check, monitor and alarm
- Common data bases and subroutines
- Computer program intercommunication

When viewed as a total operation, the space station together with its experiments is a complex on-going process and the data processing assembly appears to be best described as a real-time process-control system. Except for the generation of random or pragmatic stimuli, and the system formation of operational computer programs, the application is characteristic of automatic test and checkout systems.

2.2.1.2 Language Considerations

The advantages of using higher-order computer programming languages for spaceborne computers has received considerable attention in the past five years. The basic conclusion reached is that flight computer hardware technology has now advanced to the point where memory and real-time limitation problems are no longer critical. Prior to this situation, higher-order languages were not used because the compiler available for them did not generate efficient enough code. Another conclusion was the realization that other aspects besides code efficiency were important. The use of higher-order languages would reduce development time and labor, improve documentation, improve design modularity and allow easier program modification.

The characteristics of the onboard system of computer programs indicate that a procedure-oriented and/or problem-oriented type higher-order language should be used for most of the system. Also indicated is the possible need for an interactive user type higher-order language for crew member use.

2.2.1.3 Candidate Languages

Specification of specific higher-order programming languages for onboard programming at this stage of development is not practical. Various languages appear capable of meeting a majority of the application requirements but no language can ever meet them all. Candidate languages should include both procedure and problem-oriented languages.

Major candidate languages should include: (1) FORTRAN IV, (2) JOVIAL, (3) SPL, (4) PL/1, (5) CLASP, and (6) HAL. Comparative evaluations of most of these languages for spaceborne applications have been made and documented. A major document which provide the criteria to determine whether a higher-order language should be used for a particular application is HIGHER-ORDER LANGUAGE STUDY FOR AVIONICS PROGRAMMING by Fred D. Bostrom of the IBM Corporation. See also TM-4564/000/00, "A Comparative Evaluation of FORTRAN IV, JOVIAL, PL/1, and SPL," by System Development Corporation.

FORTRAN IV has been the major programming language used for space system programming in the past years. One version of this language has been established as the standard procedure-oriented language for all ADVANCED BALLISTIC MISSILE DEFENSE AGENCY (AHMDA) software contractors. This language has proven to be adequate for most onboard vehicle guidance and control applications. However, with the expansion of the onboard data processing functions anticipated for future space projects, new languages have been developed. Three of these new languages are:

- Space Programming Language - SPL
- Higher-Order Aerospace Programming Language - HAL
- Computer Language for Aeronautics and Space Programming - CLASP

These languages though designed primarily for onboard computers are general enough to meet the needs of production, verification, and support of most real-time applications. These new languages provide real-time control commands and signal conditioning plus common data and subroutine recognition. Their basic differences are in the approaches to the following:

- Data types of constants and variables allowed and methods of declaring them
- Data structures, i.e., allowed groupings of the above data types
- Imperative sentences and operations available between the various data types and structures
- Conditional sentences and decision-making capability to control program flow
- Repetitive operations, i.e., methods of looping
- Subroutine types and possible parameters
- General program structure, including control blocks, global and local variables and storage allocation
- Input and output statement formats

2.2.1.4 Recommended Onboard Computer Programming Language Standards

A. A single higher-order programming language should be established as a standard for onboard application and supervisory programming. This language should be selected from a list of candidate languages with special consideration given to new aerospace application languages.

B. A single interactive HOL should be established for onboard system usage. The language should require a minimum knowledge of internal configuration and programming techniques. The language should be as conversational as possible yet cover the terminology of all flight phases and experiments.

This standard language should be established and defined by a user's committee with flight personnel representation. Existing languages are available and include those listed above as well as special purpose languages developed for real-time man-machine interaction. This application area is one where consideration to development of a special language should be considered if existing languages fail to meet minimum requirements without extensive modification.

C. Highly specialized MSS onboard computer programs constrained by execution time and/or storage requirements, or by unique procedures should have the option of being programmed in assembly or machine language if efficiency benefits can be justified.

The use of this waiver from the standard language or languages should be limited to those areas of programming where extremes in efficiency can be achieved. The interface problems between the standard higher-order languages and the assembly and/or machine language must be considered.

2.2.2 Ground Facility Computer Programs

2.2.2.1 Functional Characteristics

MSS ground facility data processing involves mission and experiment management, logistics and launch support and simulation for training and acceptance. Except for the training and acceptance activity, the major function of the ground facilities will be command and control. The principal data processing requirements for the application, supervisory, and support computer programs will be:

- File management
- Symbol manipulation
- Mathematical and statistical evaluation
- Decision making
- Bookkeeping
- Timing
- Message formatting

A substantial number of large scale ground facility data processing systems, with function characteristics similar to those of the MSS program, are currently operational. Developmental and operational experience from these facilities should be directly applicable to the MSS system.

2.2.2.2 Language Considerations

A variety of higher-order computer programming languages have been used for large scale ground support data processing systems. Most of these can be classified as general-purpose, procedure-oriented languages because they allow hierarchical definitions and flexible structuring. This type of language will handle most programming problems.

Input/output operations will be a major function of the MSS ground support system. I/O expression capability should be carefully considered for any language selected. Most widely used higher-order languages have been bypassed and assembly language used when it was necessary to code I/O equipment unique to particular applications. MSS ground station requirements for I/O will place emphasis on reading, writing, and positioning of files as well as encoding and decoding data in a wide variety of external formats.

2.2.2.3 Candidate Languages

PL/1 and JOVIAL have both been widely used for large scale, space program ground support data processing. JOVIAL which is similar to FORTRAN IV, but with better data handling characteristics, has been established as a standard real-time command and control language by several Air Force and Navy projects. The availability and proven capability of both of these languages make them prime candidates for MSS ground support.

Also of prime consideration should be the three new aerospace higher-order languages: SPL, HAL, and CLASP. Many of the desirable command and control features of both PL/1 and JOVIAL have been incorporated within these new languages, making them applicable to both the onboard and the ground based phases of aerospace programming.

2.2.2.4 Recommended Ground Based Computer Programming Language Standard

A. A single higher-order programming language, compatible with or identical to that selected for onboard programming, should be established for MSS ground support data processing. Primary consideration for the standard language should be its command and control capabilities and its I/O features.

2.2.3 Support Computer Programs

2.2.3.1 Characteristics

The support computer programs are those to be used in the design, development, testing, and validation of computer programs that will be operated onboard the space station and at the various ground support facilities. These support programs can be divided into two major groups. These are: (1) language processors which consist of compilers, interpreters and assemblers, and (2) analysis and verification tools. Language processors are considered as a separate topic (see 2.3).

Analysis and verification of computer programs allow the execution of application and supervisory computer programs on large host computers. These programs are flexible in their design and allow changes in instruction sets in order to evaluate or test operational program designs. For aerospace applications these support tools have been divided into three types of simulations.

These are:

- The Interpretive Computer Simulator, which is a software system, execut es computer instructions on a large computer in simulated time, the computer functions are simulated to the "register level". It traces the data and program control flow and prints out register and memory cell contents under user control. It sends out and receives signals to and from its environment. Usually this simulator can be used for different computers of the same class. These simulators have been used extensively. Their running time is long, and multiprocessing and time-sharing are very difficult to implement efficiently.

- The Subsystems Simulator is also a software system which runs on a large computer either together with the Interpretive Computer Simulator or with an actual flight computer. It simulates in simulated time the continuous and discrete dynamics of the hardware subsystem surrounding the operation computer. For large systems with detail models, the running time will be of long duration. For mixed continuous/sampled data systems, it is difficult to obtain efficiently accurate numerical solutions.

- Integrated Computer and Subsystem Simulation is run on a large computer under interactive control by the operators. It includes an interpretive computer simulation and a subsystem simulation. The subsystem simulator uses a large number of models being described by differential and algebraic equations, transfer functions, and logical functions. These models can be described by a higher-level language which allows fast changes of models when the subsystem design changes and gives more extensive error diagnostics for the simulation set up.

2.2.3.2 Language Considerations

Analysis and verification computer programs are written in a wide variety of specialized problem and/or procedure-oriented languages. The wide variety is the result of specialized needs and requirements defined by individual software contractors. However, because the simulations are basically engineering-system oriented, the languages used are subsets or modifications of a technical language such as FORTRAN even though the contractor may establish his own name.

Because of the wide variety of languages used for analysis and verification tools, simulation results can be difficult to compare between different software contractors working on identical projects. A review and analysis of these tools for proposed MSS software contractors to determine the significance of any incompatibility should be made or standards for these tools which all contractors must meet should be established.

2.2.3.3 Candidate Languages

No specific computer programming languages are defined as candidates for the developmental and testing support of MSS operational software. Individual software contractor languages as well as support languages used for previous space projects should be considered. Below the level of subsystems, it is generally cost effective to allow software development contractors to use their individually developed support programs. For system testing and validation however, standardization of the language for the support programs should result in substantial cost and time savings.

2.2.3.4 Recommended Standards for Support Computer Programming Language

A. Languages used to program the development and testing support programs for MSS software should provide support programs which produce reasonably compatible results. This should be the case for both standardized or non-standardized languages.

B. Support program languages should be standardized for formal acceptance and validation of contract end-item software packages.

2.3 COMPILERS

Compilers are language processor development tools. In discussing their application to the MSS software system, it is assumed that higher-order programming languages will be used. Compilers operate on a host computer and generate code for a target computer. The same computer may be used as the host and as the target computer. Thus compilers can be used to generate object code for the MSS onboard and ground based operational computers with various contractor host computers. They can also be used on the operational computers themselves to enter new programs or change existing ones.

Compilers are expensive and are developed with specific features to meet specific needs. Compiler cost is dependent upon the following criteria:

Language features required, degree of optimization desired, characteristics of target and host computers, similarities of the host or target computer to a machine for which a compiler exists, type of object code to be generated, and desired options.

Compiler development has been an integral part of higher-order aerospace language development. To date, five different Space Programming Language (SPL) compilers have been developed for various host and target computers. Three different compilers are known to exist for the CLASP language and one for the HAL language. These compilers and their features are considerations in selecting one or more of the aerospace languages for the MSS.

For MSS software development, the use of common compilers by all contractors should be considered. A single compiler for both onboard and ground based programs will probably not be possible because of different machine characteristics. However, a single compiler for each could be established. This may prohibit the use of some contractor host computers but should result in overall cost and time savings because of uniformity.

Onboard compiler operation for computer program entry and modification will require the development of a new compiler. The extent to which benefits are gained by having an onboard compiler versus its development cost must be considered. With an onboard compiler the ease of entering new computer programs for changing mission phases is enhanced as is the modification or correction of programs via ground relayed instructions. The extent to which these features are desirable will require additional study.

Basic standards and conventions for MSS compilers should include the following:

Common compilers for the development of onboard and ground based operational computer programs should be established.

Existing compilers should be used wherever possible for the MSS program.

If an onboard compiler is developed, it should have extensive test and debug features.

2.4 PROGRAM PRODUCTION

The magnitude and variety of computer programs required for the MSS software system will require the use of a number of contractors. It is very unlikely that a single organization could produce all programs within the time allowed for acquisition. To aid compatibility and configuration control of programs produced by different sources, software production standards and conventions are required.

2.4.1 Libraries

Libraries of subroutines and algorithms as a technique for computer program organization are discussed under program organization. For a modularized software system of the degree proposed for the MSS system, the normal concept of computer program libraries is applicable to each level of subprograms. Because any given subprogram can theoretically call and be operated with all subprograms in lower levels, these subprograms consitute a library for that subprogram. To assure operation, standards for compatibility must be consistent for all subprograms within the system.

In producing the operational application and executive computer programs, a contractor may employ a library of subroutines to conduct checkout and testing operations. Standardization of these libraries between contractors is not generally cost effective because of the different source computers used. However, a centralized library maintained by an integration contractor and available to all contractors has proven to be a valuable technique.

Libraries of subroutines and algorithms used for checkout and testing shall not require operational modifications to the programs being tested.

Each subroutine and algorithm library used to produce operational computer programs shall be documented with standards of acceptance and content.

2.4.2 Production Monitoring and Quality Control

Production process monitoring and quality control are management functions performed at the contractor and overall project levels. Various PERT and milestone type systems are the basic tools used for monitoring with the parameters of running, coding, and debugging times as measurements of quality. Monitoring and quality control systems to be employed for the MSS program should be defined by project management and specified as contract requirement for all contractors.

A standard production monitoring and reporting system should be used by all MSS software contractors.

2.4.3 Program Segmentation

Specific program production segmentation standards and conventions can only be established after the organizational structure of the programs have been defined. Standards and conventions for this technique should be specified following the publication of MSS computer program design requirements.

2.4.4 Interpretive Simulations

A major problem in previous aerospace software program production has been the variety of allowable interpretive simulations. In most cases the cost and limited availability of actual flight computer hardware, and other systems, required each contractor to simulate the characteristics needed for software checkout and testing. Differences in these simulations between contractors resulted in inconsistent results for identical software elements. MSS software management should define simulation standards and conventions for those contractors required to perform simulations or provide one or more standard simulation facilities for contractor use.

Simulation standards for the operational environment, and space operated hardware should be defined for all simulations to be used for software production checkout and testing.

2.4.5 FIX Programs

Standards and conventions for the application of FIX programs are program specific and dependent upon the design of the FIX programs. If this technique is utilized in MSS program production, rules of application for the specific programs used will be generated as standards and conventions.

2.4.6 Machine Independent Programming

Computer programming language and compiler standardization will in general establish the standards and conventions for this technique. It is now common practice to produce target computer programs on a variety of host computers by employing compilers. It is assumed that compiler technique will be employed in the MSS program.

2.4.7 COMPOOL (Data Dictionaries)

For the modularized MSS software system the use of data dictionaries for both onboard and ground based processing will provide additional flexibility. Modularization will require the separation of program instructions and data. If the data can be structured on some common basis such as: application, frequency of use, experiment, operating mode, etc., a data dictionary will permit rapid and repeated use. The use of this technique requires the establishment of standards for naming and organizing the data files to be addressed through the COMPOOL. Additional study is required to determine the degree of COMPOOL application to the MSS system. Therefore, no generalized standards are presented. A substantial list of COMPOOL standards and conventions have been defined for those systems using the technique.

2.4.8 Allocation Programs

Standards and conventions for the application of allocation techniques in the production phase require a detailed design of the organizational structure of the software system. Following the establishment of MSS-ISS memory design and operating specifications, allocation standards can be defined.

2.4.9 Flow Chart Generation

A standardized automatic flow chart generating computer program should be utilized by all MSS software contractors.

2.4.10 Initialization

For all computer program testing and checkout, standardized initial condition values should be used. Standardized initial conditions for each MSS operational phase should be defined.

2.5 PROGRAM ORGANIZATION

Prior to the initiation of any actual programming, software developers must have detailed organization and design information relating the new programs to the software environment, the hardware environment, the monitoring and control philosophy and the implementation procedures. This information is usually produced as a Software Organization and Detailed Specification document which has been developed from software capability and design specifications. This document contains the standards and conventions to be used in the organization of all computer programs.

Software system organization and detailed design specification will be a major effort for the MSS project. This will be particularly true for the onboard portion of the system where modularity and multiprocessing will be utilized.

2.5.1 Modularization

Modularization refers to the organization of operational computer programs on a total systems level. The concept involves the separation of computer programs into a structure of subprograms, and data files into tables which contain both data files and control instructions. Subprograms are structured and classified by a criteria which essentially defines main subprograms and a hierarchical level of other subprograms which may be called in various orders to meet system needs. Operationally, a subprogram obtains data from a given table data file, processes this data as instructed by the control indicators of the table and outputs processed information to other and/or the same data file. During the execution of any subprogram, transfer to a next lower level subprogram, with return, can be exercised by exit conditions in that subprogram.

Essentially, modularization involves the structuring of a single or basic set of main subprograms, which will be operated to meet system requirements, and a set of subroutines to be used with them. The structure is progressive in the sense that the subroutines for the main set of subprograms can be considered as a second level of main programs and a set of subroutines used in their operation will be established for them. This process is continued to as many levels as is necessary to meet total system requirements. By organizing computer programs in this manner, a software package of computer programs required to accomplish any system function can be formulated from the subprogram structure.

Modular organization provides a software system more amenable to change and modification because changes are made at the subprogram level. This allows a more systematic and less expensive verification process. It also improves system software reliability by reducing complex software structures to simple subprograms whose operation can be better visualized and can be produced more error free.

Software modularization requires standardized design rules. Conceptually all subprograms must be capable of being operated with any other subprogram. In this respect the executive control software and the interface requirement between subprograms is much more stringent than in non-modularized systems. To achieve modularization when the development and integration work is distributed over a large number of contractors, uniform design and interface definitions are essential.

All MSS software contractors will conform to the computer program organization standards and conventions as specified in detailed design specification and configuration control procedures.

Entry to any modularized subprogram must be via both a subprogram name and a numerical number which unambiguously identifies the subprogram and its operational level.

Variations in accuracy requirements and minor modifications of technique in a function performed by a subprogram should be organized as next lower level subprograms wherever possible. Duplication of subprograms with similar functions should be minimized.

2.5.2 Executive and Monitor Programs

When software modularization is combined with a multiprocessing capability, organization requirements for executive and monitor software is compounded. Unless each processor of the multiprocessing system has an independent set of subprograms, a scheduling procedure to handle simultaneous request for subprograms must be defined. This may be overcome by assigning processors computational tasks which will not result in a subprogram conflict. An executive control system to provide this type of scheduling could become very complex. To fully utilize the benefits of a combined modularized software system and a multiprocessing capability, extensive trade-off studies for executive control will be required. These trade-off studies will provide the standards for executive program organization.

Standardized alarm and display routing should be defined within the executive system. An afterthought of many large ground support systems have been the user alarm messages and displays which indicate a degraded or degrading system. A standard distribution for each alarm signal, with priorities, should be established.

Standards and conventions of the executive and monitor software system will be defined as operating rules for the system. Operator and user interfaces will be specified in terms of allowable operations and output.

A priority for both critical and non-critical operations and alarms will exist within the executive system.

The executive will assume responsibility for saving all guaranteed data between subprograms. This feature is not to be incorporated within the subprograms.

2.5.3 Libraries

Libraries of subroutines and algorithms have been used extensively on non-modularized software systems. They represent the first step of modularization. If total modularization of the MSS software is not achieved, libraries should be considered as a next level alternative. Compatibility of libraries with main programs is required but because many library items will be limited to operations with specific computer programs, their compatibility requirements are less than in a modularized system.

A call-by-name procedure shall be used for all subroutine libraries.

Standard mathematical and data processing library subroutines should be used in preference to similar functions derived from other sources.

2.5.4 Segmentation and Allocation

Segementation and allocation as used in this discussion refers to the organization of an individual computer program. Its importance is in producing efficient computer programs. For the MSS modularized software, this organization activity could be highly significant. All subprograms should be organized to

perform their function within reasonable time limits and with a minimum of executive control. A sequence of subprograms requiring excessive computation time and control may fail to meet total system requirements.

Individual computer programs will be optimized for running time and/or control requirements.

Individual computer programs will be organized in a logical sequence consistent with the modularization concept in order to expedite any required modifications.

2.5.5 MACRO Programming

Programming with MACRO instruction refers to use of an instruction that represents several simpler commands. A language capable of accepting MACRO instructions provides a programmer with a means of extending the language to mechanize complex but repetitive operations. For programs which use a subroutine several times a MACRO may be coded into the final program each time it is required but the subroutine need only exist once in the object program.

For a modularized software system the use of MACROs in the subprograms will enhance the organization of these programs. Their greatest benefit will be in those portions of the subprograms that are less subject to change or modification.

Standards and conventions for MACRO use are required to document and communicate their application. One of the greatest difficulties experienced with their use is the problem of a different programmer learning someone else's MACROs.

2.6 FORMATS

2.6.1 Documentation Format Standards

The standardization of document types, outlines and contents are a part of Configuration Control. (See Documentation) Specific formats for these documents are specified in the Standards and Conventions document. Basic format items are illustrated by the following:

All computer program documentation will be published on 8 1/2 x 11-inch stock. Fold-out sheets should be avoided wherever possible.

Company or organization name, document number, authors, computer program name (including modification number), date of publication, and other pertinent document identification information shall be included on the document cover as well as the title page.

Flowchart descriptions and symbols will conform to established project standards with additions or variations noted by an asterisk.

Documentation formats not specified shall follow good publication conventions.

2.6.2 Message Format Standards

Message transmission is primarily an input/output operation controlled by a supervisory system. Message formatting is a computer programming task and is usually performed as a part of an application program or by an operational support program specifically designed for this task for a group of application programs. Message transmission and formatting will be a major activity for the MSS Information Management subsystem. Interfaces between the space station vehicle, the space shuttle, ground support facilities and the proposed modularity and multiprocessing organization of the DPA will be the focal points of this activity.

Message format standards and conventions which should be met by all system meassages are as follows:

Operator input formats and contents should be made in a language understandable to the user.

Coded abbreviations requiring the consulting of documentation for meaning are not permitted.

Identical message formats will be used for messages generated by the same event.

All messages should be concise and unambiguous.

2.6.3 Data Card Standards

Symbolic data cards are used for source program statements, data specification statements, and input data. There are three basic types of formats used in their application. These are: free-field, fixed-field, and a combination of the two. In general, cards designed in a free-field format are a convenience to the user but require more processing. The fixed-field format can be processed more efficiently by a program but is more difficult to script.

The major considerations in establishing standards and conventions for card formats are that free-field cards are recommended for control options while fixed-field cards are most commonly used for input data. Another aspect of data card formatting is the organization and identification of a card deck. For uniformity of operation and documentation, standards for structure are also required.

All programs which process free-field data should use a centralized free-field format to simplify the decoding operation.

A standard language and format for free-field data should be established for consistency, simple usage and to minimize errors.

A standard end-of-record or end-of-file card should be used to signify sets of data cards or end of data.

For a fixed-field format special card stock should be designed.

2.6.4 Magnetic Tape Standards

In most cases, contractor developed source programs and data will be delivered on magnetic tape. Standards and conventions for the physical properties and content of these tapes are required for configuration control. The following tape standards and conventions are recommended for the MSS software system:

All deliverable magnetic tapes shall be nine channels and 1600 bits per inch.

When more than one magnetic tape is required, the last magnetic tape should contain specific end-of-file marks after the final record.

Each deliverable tape shall be identified by a description file which shall contain: (1) contractor's name, (2) contract number, (3) date, (4) project title, (5) number of logic file, (6) number of physical files, and (7) keypunch code used.

Tapes shall not contain header or trailer records.

2.6.5 Display Standards

Onboard and ground station displays will be extensive for the MSS program. Factors which must be considered in obtaining and displaying information for human use are equipment capabilities, information availability, human capabilities and system processing logic. Standards for display format design are required to enhance the ease and efficiency of system operation.

Display formats shall be met the following basic criteria:

- Easily read
- Can be read accurately by operator
- Data value changes easy to detect
- Directly usable form
- Display failure immediately apparent
- Alarm and status apparent
- Messages attract attention
- Use standardized symbols

2.7 TERMINOLOGY AND ABBREVIATIONS

The magnitude and variety of modular space station operations and experiments will require substantial terminology and abbreviations for documentation and computer programming. The necessity for establishing standards to provide common meaning is obvious. However, because of the magnitude of the terminology and abbreviations that will be employed, consideration should be given to segmenting the material.

Standard terminology and abbreviations to be used by all MSS computer program system contractors should be defined, published and maintained by an authorized organization.

Standard computer program terminology and abbreviations should be subdivided into groups or categories on a basis of user needs. By subdividing on the basis of major operations or functions, the task of publishing and maintaining standard terminology and abbreviations may be substantially reduced. The objectives in subdivision should be to minimize the number of different lists required yet meet the needs of all users. Suggested subdivision categories include:

- By MSS Operation
- By MSS Experiments
- By Onboard ISS Functions
- By Ground ISS Functions
- By DPA Modules

All standard terminology and abbreviations established should be specific and unambiguous for their meaning throughout the system. This is an obvious standard yet its failure to be followed has resulted in computer program design, development, and operational problems.

2.8 SYMBOLS

Like terminology and abbreviations, a large number of symbols will be required for the computer programs, displays and documentation of the MSS software system. Technically acceptable alphabetic, numeric, and geometric symbols are limited and have different meanings in different generic disciplines. Because of the large number of technical disciplines to be employed in the MSS system, symbol standardization will require considerable organization and effort.

Standard, unambiguous, MSS software system symbols should be defined, published and maintained by an authorized organization. All scientific, mathematical, engineering, and special symbols used to design, produce, test, document, and operate the software system should be standardized. Computer program flowchart symbols are a prime example of what should be standardized. Flowchart symbols such as those defined by the United States of America Standards Institute (USASI) and other acceptable organizations are well documented and have been accepted and included without exception in standards and conventions documents.

Symbols selected as standards should represent a best compromise between their generic discipline of use, the programming language limitations in which they will be used, and their reproducability by printing and display equipment. Conflicts in symbol meanings between disciplines will require adjustments and modifications in order to standardize. The possible impact of such adjustments on other system parameters must be considered. Complex symbols difficult to construct and reproduce should be avoided.

Display symbols should be considered in respect to their human engineering aspects as well as their defined meanings. Symbols which can be easily read and not misidentified are important for efficient user interaction and interpretation.

Provision for specialized symbols should be provided in the standardization procedure, but should be limited and controlled. Requirements for specialized symbols are highly probable for some of the experiments to be conducted by the MSS. When the use or establishment of such specialization is justified, because of user needs, it should be provided.

2.9 UNITS OF MEASUREMENT AND CONVERSION

Standard units of measurement and conversion are required for mathematical consistency within and between computer programs. These standards are usually defined within the computer program design specification documents for each computer program to be developed and are listed in the standards and conventions document.

Standardized units of measurement, conversion factors, and coordinate systems for space vehicle operations appear to be well defined and relatively uniform between projects. For MSS mission operations standardization of measurement and conversion units can be achieved by the direct application of previous experience. MSS experiments, however, will probably require a specialized set of measurement and conversion units. The extent to which they will be required will depend on the level of real-time data reduction required for the experiment.

Total standardization of units and conversion factors for MSS operations and experiments should be an objective, but allowance for specialization must be provided.

Units of measurements used with all data defined in the system data base should be based on the MKS system. This should include a listing of basic units such as: length, mass and time, and derived units such as: area, volume, density, acceleration, etc.

Specialized units of measurement, conversion factors and coordinate systems for specific experiments should be allowed when justified to meet mission objectives. Control and integration of these specialized factors into the total system should be accomplished through a users committee or organization.

Conversion factors must be compatible with the computation ability and accuracy of the system computers. Data word size plus fixed and floating point computational ability are of major consideration.

All coordinate systems used within the system should be defined by their location of origin, primary plane, and principal direction.

2.10 DOCUMENTATION

Standard outlines and content descriptions of all technical documents required to research, design, develop, test and operate the software packages for the MSS program should be specified. This requirement will normally be established in the contract end item description of Configuration Control

Management procedures. It is assumed that some type of configuration management such as AFSCF Exhibit 375-2 or that employed for the Apollo Program will be applied to the MSS program.

2.10.1 Standards

Standardized outlines and content descriptions of all technical documents required to support all MSS software development functions shall be defined and published as a part of MSS configuration management.

Software documentation shall include but not be limited to the following:

- Software development requirements
- Software capability description
- Software design specification
- Software organization and detailed specification
- Configuration management plan
- Testing and evaluation procedures
- Installation procedures
- Non-technical general description
- Programmers manual
- Operators manual
- Users manual
- Maintenance manual
- Standards manual

Flow diagrams for computer programming documents should follow the following standards:

- Fit into a 6-1/2-inch wide by 8-inch long area on a 8-1/2 by 11-inch sheet of paper. Foldout sheets should be avoided wherever possible.

- The flow diagram should run from the top of the page to the bottom.

- Acronyms and words outside of a block should be capitalized.

- Standard symbols must be used.

2.10.2 Conventions

In the illustration of an operation, the program parameters should be presented by mnemonics which are suggestive.

Each format input and output parameter should include: type of parameter, class of parameter, type of item or array, and a description of what the parameter is to contain.

Zero bit position should be specified for descriptions of computer words.

2.11 TESTING AND VALIDATION

Standards and conventions for testing and validation are generally considered and specified as part of configuration management rather than a part of a standards and conventions document. Most configuration management systems specify testing and validation standards through a series of three documents. These are: (1) the test plan, (2) test procedures, and (3) test results. Additional structure is established by defining testing categories for component, subsystem and system testing.

Test plans, procedures, and the reporting of results should be standardized for all MSS software by a configuration management system.

Standardized testing tools should be utilized by all software contractors wherever possible. Specification and descriptions of these tools should be defined in Equipment and Program Requirements section of each test plan. Standardization of the following testing tools should be considered for the MSS program:

- Vehicle, environmental and onboard computer simulations.
- Ground computer simulation.
- Tape or card inputs from interfacing systems.
- A computer programming testing language for systems integration.

2.12 MAINTENANCE

Standards and conventions for computer programs maintenance, like testing and validations, are primarily configuration management-specified items. To maintain control of an existing operating system, extensive review and implementation procedures are required to assure compatibility of any proposed changes or modifications. One area where standardization must be established is in the identification and description of changes made to the system. Maintenance procedures, implementation, and standards should be defined in detail in the configuration management plan.

2.13 MANAGEMENT

The management of computer program standards and conventions is established and administered through a configuration management process. Those standards and conventions required for the design, organization, and development of source programs and data are contained in a STANDARDS AND CONVENTIONS DOCUMENT, while those required for documentation, validation and testing and maintenance are specified as specific items in the CONFIGURATION MANAGEMENT PLAN.

Computer program standards and conventions documentation should be established as a contract end-item for the computer program integrating contractor or equivalent organization. This documentation should be made available to all contractors with the understanding that its content is to be followed unless authorized deviation is requested and granted.

Maintenance of computer program standards and conventions documentation should be specified as an on-going task.

Configuration control procedures should define all organizations and methodology for the management of project standards and conventions.

Table 2-1. Baseline Standards and Conventions

COMPUTER PROGRAMMING LANGUAGE	A higher-order programming language (SPL or HAL) should be used as the standard higher-order programming language for all MSS on-board, ground and supporting computer programs.
COMPILERS	Existing HOL compilers modified to meet any special MSS system requirements, if any, should be established as standard host developmental compilers to be used by all software contractors. New HOL compiler development should be limited to the production and organization of code for the on-board and ground based target computers.
PROGRAM PRODUCTION Libraries	A computer program integrating contractor, or an equivalent organization, shall establish and maintain a computer program library of all accepted computer programs, subroutines and algorithms. Standardized configuration control identification procedures shall be employed for each item contained in the MSS software system library. All MSS software system contractors shall have access to MSS software system library items for checkout and testing purposes. All library items shall be written in the selected HOL and shall be documented by: (1) a general non-technical description, (2) a capability description, (3) a test acceptance and condition report and (4) a user's manual.
Production Monitoring and Quality Control	All MSS software contractors shall follow production status reporting procedures and schedules as defined by Software Configuration Management Requirements. All MSS software contractor conducted assembly, subsystem and system testing shall conform to established MSS system test plans and the results documented in a standardized format as defined by Software Configuration Management Requirements.

Table 2-1. Baseline Standards and Conventions (Continued)

PROGRAM PRODUCTION (continued)	
Program Segmentation	No specific MSS software production segmentation standards and conventions are recommended at this time.
Interpretive Simulations	All MSS software contractor conducted Interpretive Simulations shall conform to standardized MSS hardware and environmental characteristic equations. Contractor conducted Interpretive Simualtions shall be operationally compatible with subsystem and system computer program integration simulations to be conducted by the same or a different software contractor.
FIX Programs	Automated FIX computer programs for hardware and software errors are not recommended for MSS-ISS computer program production.
Machine Independent Programming	No standards and convention related to machine independent programming are required if a HOL is used.
COMPOOL (Data Dictionaries)	The primary criterion for grouping data elements should be frequency of use. Wherever possible, the data should be structured by the following: (1) read-only data, (2) read/write data, (3) multiple-record data and (4) single record data. Large blocks of related data should be structured as a single file with multiple records. Each record should have the same structure to allow a single COMPOOL definition to access the file one record at a time.
Allocation Programs	SPL Table Allocation, Programmer Allocation and Compiler Allocation rules and recommended procedures shall constitute allocation standards and conventions and should be followed by all software contractors.
Flow Chart Generation	The IBM automatic flow chart generating program FLOWCHART, or an equivalent, should be established as a standard for the production of all MSS computer program flow charts.

Table 2-1. Baseline Standards and Conventions (Continued)

Initialization	Computer program production testing and checkout shall be performed with standardized initial hardware and environmental conditions. Standardized system equations and initial parameter values for all MSS operational phases shall be defined for software system and subsystem testing.
PROGRAM ORGANIZATION	
Modularization	No recommendation at this time.
Executive and Monitor Programs	No recommendation at this time.
Libraries	No recommendation at this time.
Segmentation and Allocation	All HOL written computer programs shall be organized as follows to simplify maintenance, documentation and checkout: . START . Description Comments . Declarative Statements . Imperative Statements . Global Closes . Procedures/Functions . TERM
MACRO Programming	MACRO programming as a separate technique for computer program organization is not recommended.
FORMATS	
Documentation	See para. 2.6
Messages	The selected HOL language standards and conventions for INPUT and OUTPUT operations and the manipulation of TEXTUAL Communications shall be followed for all message formats. The general recommendations of para. 2.6.2 should be established as message format conventions.

Table 2-1. Baseline Standards and Conventions (Continued)

Data Cards	Free-field formatting should be established as the standard for all symbolic data cards. Recommended conventions for data card formats consistent with the HOL usage should be followed.
Magnetic Tape	See para. 2.6
Displays	See para. 2.6
TERMINOLOGY AND ABBREVIATIONS	See para. 2.7
SYMBOLS	See para. 2.8
UNITS OF MEASUREMENT	See para. 2.9
DOCUMENTATION	See para. 2.10 Software documentation standards and conventions as defined and specified in MSS software configuration management procedures shall be followed by all software contractors. MSS software configuration management publications shall define the types of software documentation required for all computer programs. These publications shall also define a standard outline and a content description for each required document.
TESTING AND VALIDATION	Standards and conventions for computer program testing and validation, at the assembly, subsystem and system levels shall be defined as test plan requirements by configuration management procedures. All computer program testing and validation shall be documented with: (1) Test Plan, (2) Test Procedures and (3) Test Result documentation.

3.0 COMPUTER PROGRAM SPECIFICATION TREE

3.0 COMPUTER PROGRAM SPECIFICATION TREE

The computer program system specification hierarchy (tree) is defined. In addition, a preliminary top level computer program system specification was prepared which will identify all computer program documents and tapes in accordance with the standards recommended.

This task was accomplished in four steps.

Analyze Existing Specification Schemes/MSS Needs. The computer specification schemes currently in use were analyzed in terms of the needs of the MSS. Identification, concept, advantages, and disadvantages of each scheme are presented in narrative and tabular form.

Evaluate Schemes. The computer program specification and hierarchy schemes were evaluated against the needs of the MSS program requirements. The MSS scheme will be required to handle a problem unique to the Space Station and quite different from ground applications. The types of personnel utilizing documents and associated materials will be primarily engineers and scientists as opposed to programmers. The result of this step will be a recommended scheme to be implemented for the Space Station Program.

Develop Specification Hierarchy/Component Description. Utilizing the results of the first two steps, the hierarchy of documents and tapes required for the Space Station Data Processing System was specified. Each component represented in the hierarchical structure was described in detail; e.g., the outline and contents of each type of document was given.

Develop Top-Level Specification. Working from the MSS programming requirements and the basic concepts, a top-level specification containing system performance and design requirements, system segment allocations and interfaces, identification of major contract end items, and requirements for system testing was produced.

The objective of this study was to:

a. Develop a computer program specification hierarchy, and to describe in detail each component in the hierarchy.

b. Prepare a preliminary top level computer program system specification, containing system performance and design requirements, system segment allocations and interfaces, identification of major contract end items and requirements for system testing.

This specification establishes the top level requirements for the performance, design, test and qualification of a computer program identified as the Modular Space Station Master Computer Program and defines in logical, and

operational language the detail necessary to design, produce, and test this program. Data processing functions from all other subsystems are included in this specification.

Further studies need to be done to determine the requirements specified as TBD.

3.1 HIERARCHY OF PLANS AND SPECIFICATIONS

The following paragraphs describe the hierarchy of the plans and specifications.

Specification Hierarchy

Figure 3-1 illustrates the logical sequence of Contract End Items to be produced and delivered, and the responsible agency.

Detailed Specification Tree

Figure 3-2, page 3-9, illustrates the detailed Specification Tree containing each individual specification as part of the DPA. For clarity the computer program/modules listings have been indicated by symbol (A). Each symbol represents a number of listings corresponding to the number of subprograms or modules contained in the Part II Specification.

Specification Format

Figure 3-3 on page 3-11 illustrates a sample containing the minimum information that is to be supplied on the cover sheet of any deliverable specification. The specification number and CEI number will adhere to the numbering method developed in the Configuration Management Specification.

3.2 SPECIFICATION DESCRIPTION

In the paragraphs below are described the Computer Program Specifications and specification type documents that will constitute the basis for the software segment of the Modular Space Station Program. An identification method for all specifications and specification type documents will be developed in the Configuration Management Specification.

Responsible agencies are indicated for each specification. A Software Integration Control Agency, also referred to as Computer Program Integration Control (CPIC) needs to be established as a responsible agency for some specifications as indicated.

The Space Station Computer Program Development Plan will be produced by the Computer Program Integration Control Agency (CPIC). It will specify the plan definition consisting of the requirements for monitoring the design and development, the quality control and standardization of computer programs and the contractor responsibilities. It will set up schedules, establish management control and progress reporting procedures and requirements. This plan will be one of the basic tools used by the Integration Control Agency for management control of all space station computer programs.

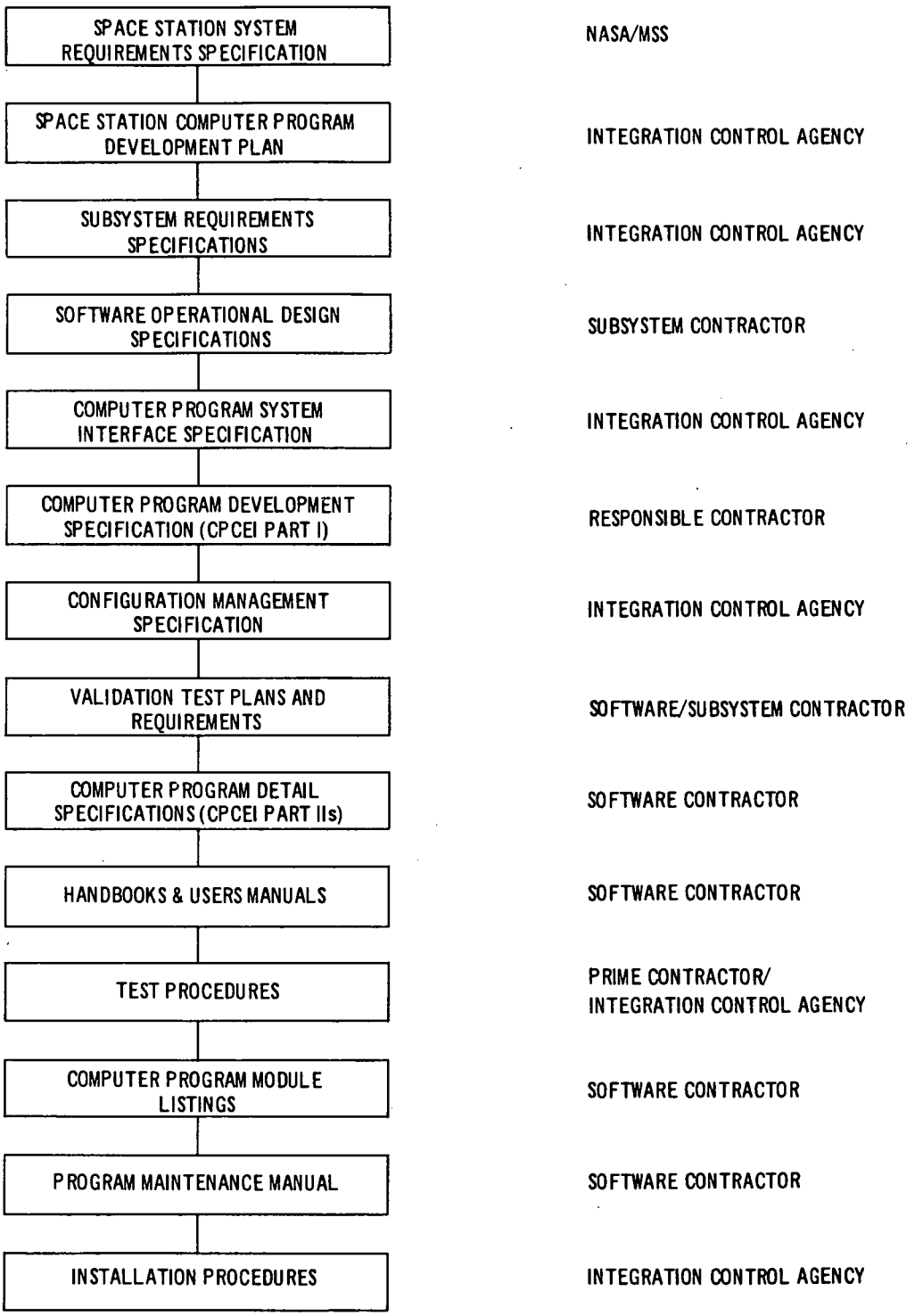


Figure 3-1. Hierarchy of Plans and Specifications for MSS

The Subsystem Requirements Specification will be produced by the Computer Program Integration Control Agency. It will specify the requirements allocated to the Data Processing Assembly for each of the seven subsystems, viz.:

Guidance and Control Subsystems
Electrical Power Subsystem
Reaction Control Subsystem
Environmental Control and Life Support Subsystem
Information Subsystem
Crew Subsystem
Structures Subsystem

The subsystem contractor will respond to the Subsystem Requirements Specification with the Software Operational Design Specification, which will use the Subsystem Requirements Specification as the basis for its design.

The Software Operational Design Specification will be produced by the Subsystem Contractor. It will specify the requirements and design of the data processing functional requirements for each of the subsystem's components based on the Subsystem Requirements Specification compiled by the Integration Control Agency. It will give system limits, data base requirements, interface requirements, data rates, quality assurance provisions and checkout procedures.

A Software Operational Design Specification is required for each of the subsystems, except for the Information Subsystem (ISS). For ISS the required design will be included in the Computer Program Development Specification (CPCEI Part I).

These Software Operational Design Specifications will form the basis for the "Subsystems Operations Management" program of the applications programs.

The Computer Program System Interface Specification will be produced by the contractor responsible for the development and design of the software package. It will specify the functional requirements, the system limits, data base requirements, human performance requirements, interface requirements, and programming language, standards and conventions. In addition it will specify the quality assurance provisions, modularity and commonality, message formats, displays, alarms, and manual control entries.

A CPCEI Part I Specification is required for each set of program assemblies such as supervisory and application programs, support programs and data base.

The Configuration Management Specification will be produced by the Computer Program Integration Control Agency. It will specify the Configuration Management Program Objectives, authorities and responsibilities. It will establish specification and documentation requirements (baseline) control, procedures and requirements for change control, define classes and types of changes and maintain configuration status control.

The Validation Test Plans and Requirements will be produced by the contractor responsible for the development and design of the appropriate software package.

It establishes the detailed requirements, criteria, general methods, responsibilities, computer programming tools and equipment and overall planning for the validation test plans for qualification of the Modular Space Station Computer Programs' functional grouping.

Testing will be conducted in accordance with the Quality Assurance provisions of the applicable Computer Program Development Specification (CPCEI Part I) and this test plan will verify that the CPCEI's perform in accordance with the requirements as stated in the particular Part I Specification.

The Computer Program Detail Specification (CPCEI Part II) is produced by the contractor responsible for production of the computer program.

This document describes the programmed functions, how programmed functions are entered, and their purpose. The CPCEI Part II Specification will provide a detailed flow chart of the computer program, a description of the interfaces of the program and other computer programs, and the data organization (files, tables, items) and constants used. Computer program limitations are also specified.

A Computer Program Detail Specification will be produced for each defined computer program.

A Handbook for each position in the MSS is prepared by the contractor responsible for the software design and development. Purpose of the Handbook is to describe in detail the human interaction with the onboard data processing system as well as with the ground-based data processing facilities, the detached modules and the Space Shuttle. Procedures, responsibilities, duties, and emergency measures are specified.

The User's Manual is produced by the contractor responsible for the design and development of the particular program system. Purpose of the User's Manual is to provide the user with a prose non-technical description, enabling the user to operate the particular system. It provides operating instructions for start and control of the system, all detailed inputs to the particular system, all outputs, and instructions for maintenance. User's Manuals are produced for each individual utility and support data processing subsystem, such as Simulator, Data Reduction.

The Test Procedures are produced by the contractor responsible for the testing of the DPA in conjunction with NASA. The purpose for this document is to outline and to detail the procedures to be followed in testing; to start, maintain, terminate and restart the program; the manning required and responsibilities, location and schedules necessary for the testing.

The Computer Program Module Listing is produced by the responsible software contractor. It will give:

a. The source program module listing containing the higher order language symbolic inputs to the compilation of the program

b. The generated machine code instructions of the program, listed side by side with the octal or hexadecimal representation

c. A memory map indicating the allocation of every register of operating memory as specified by the program module's dictionary,

d. A dictionary

e. An index register usage table, indicating register use as system index, special index or neither

A Computer Program Module Listing is required for each module, such as System Clock Control, Queued Task Status Control, Allocator, Display Formatting, etc.

The Program Maintenance Manual is produced by the Contractor responsible for the design and development of the MSS Software.

Its purpose is to provide the user with a prose document in non-technical language describing the program functions, program operations, its inputs, outputs, and limitations, and any malfunction conditions, that can occur, with the programmed response.

Installation Procedures are produced by the Computer Program Integration Control Agency.

Purpose of this document is to provide a uniformly standardized procedure for installation of all contractually produced software.

3.2 COMPUTER PROGRAM TAPES

Computer program systems and modules will be delivered on multitrack magnetic high-density tape, disk or disk pack. High-density magnetic tape provides a significant storage advantage over low density tape or disk. The tape or disk contents will consist of one or more files.

3.3 TAPE IDENTIFICATION

A tape or disk will be identified by its first File Ident. This File Indent shall be as a minimum consisting of the following:

a. System/Equipment Code
b. Contractor Code
c. CEI Number
d. Serial Number
e. File Protect Key
f. Record Type Code
g. Number of Files on Tape

The tape must be physically labeled with at least the information specified under a, b, c, and g.

3.4 FILE ORGANIZATION AND STRUCTURE

Files on tape or disks will be structured as follows:

a. File Ident
b. Data Records
c. End of File Record

There can be more than one file on a tape or disk.

3.5 TAPES REQUIRED AS CEI

The following tapes will be required to perform all data processing functions for the MSS.

- Subsystems Computer Program Tape
  compiled from the programs of each subsystem, viz.
  - Guidance and Control Subsystem
  - Electrical Power Subsystem
  - Environmental Control & Life Support Subsystem
  - Crew Subsystem
  - Structures Subsystem
  - Communications Subsystem

- MSS Master Computer Program Tape
  containing:
  - Supervisory Programs
  - Application Programs
  - Application Support Programs
  - Data Base

- Support Program Tapes
  - Simulator Programs Tape
  - Utility Programs Tape

- Test and Validation Computer Program Tape
  containing:
  - Test and Validation Programs

3.6 PRELIMINARY TOP-LEVEL COMPUTER PROGRAM SYSTEM SPECIFICATION

The remainder of this section is presented in CPCEI Specification format, describing: System Performance and Design Requirements, System Segment Allocations, Interfaces, Major Contract End Items, and System Testing Requirements.

Page Intentionally Left Blank

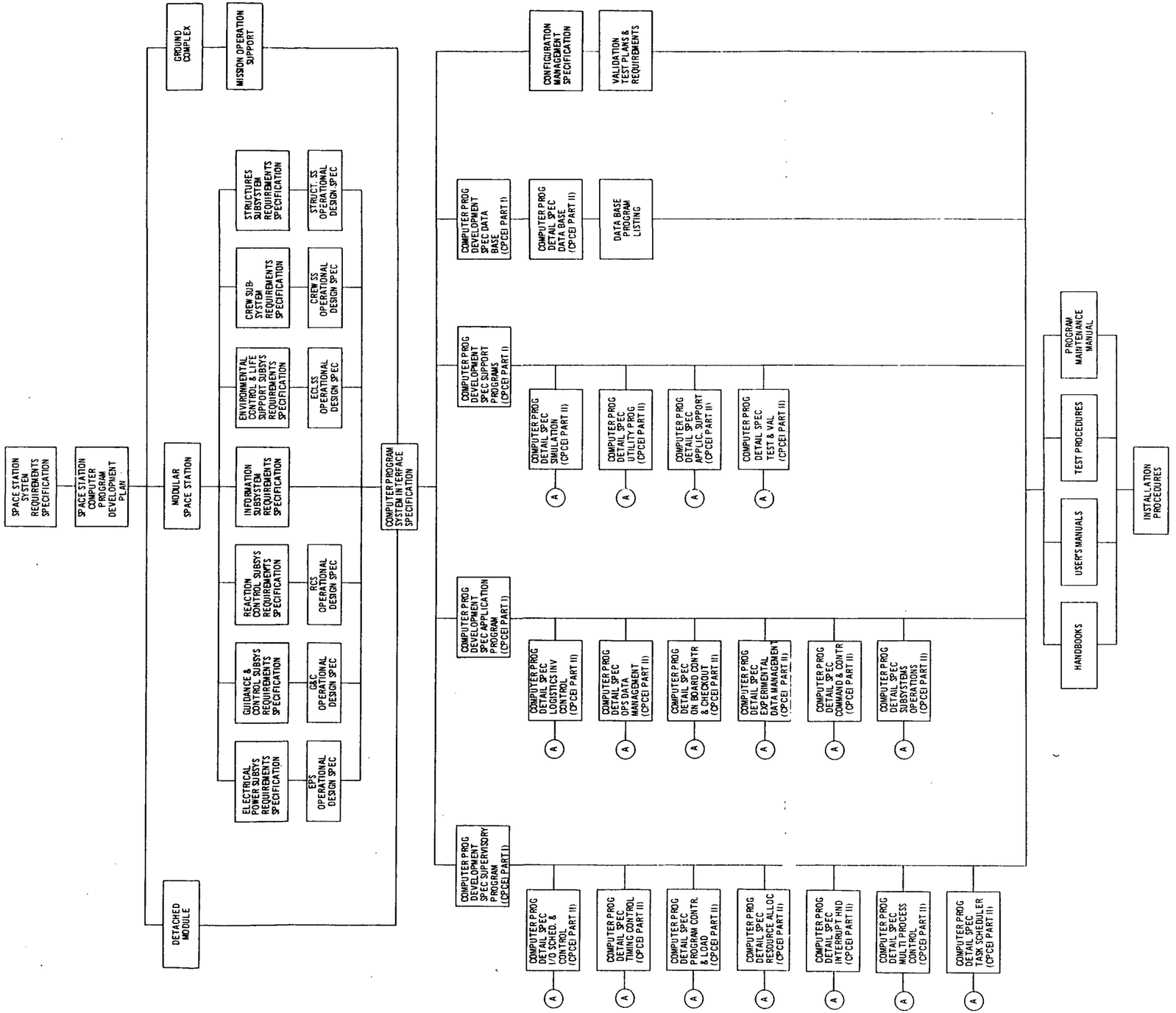


Figure 3-2. Detailed Specification Tree

Page Intentionally Left Blank

Specification No. _____

Release Date _____

CONTRACT END ITEM DETAIL SPECIFICATION

(COMPUTER PROGRAM)

PERFORMANCE/DESIGN
AND
PRODUCT CONFIGURATION
REQUIREMENTS

(CEI NUMBER)

(Approved Nomenclature)
for
(System Name) (System)

Approved by: _____
(Contracting Agency)

Approved by: _____
(NASA/MSC)

Date: _____

Approval Date: _____

Contract Number: _____

Figure 3-3. SAMPLE FORMAT FOR SOFTWARE SPECIFICATIONS

3.6.1 INTRODUCTION

3.6.1.1 Purpose

The purpose of this part of the document is to specify the top level data processing requirements and the computer program components of the Data Processing Assembly (DPA) of the Modular Space Station (MSS) program.

3.6.1.2 Scope

This specification establishes the top level requirements for the performance, design, test and qualification of a computer program identified as the Modular Space Station Master Computer Program and defines in logical and operational language the detail necessary to design, produce, and test this program. Data processing functions from all other subsystems are included in this specification by means of the Subsystems Operations Management Program Component.

Figure 3-4 illustrates the major computer program component hierarchy for the Space Station Program.

3.6.2 REQUIREMENTS

Performance, interface, and design requirements specified herein shall be applicable to the Modular Space Station Master Computer Program.

3.6.2.1 Performance

MSS will be used by NASA/MSC:

a. To establish a centralized and general purpose manned laboratory in earth orbit through modular station entities

b. To conduct and support scientific and technological experiments

c. To develop further space exploration capabilities

The system, operational, data base, and human performance requirements specified herein shall establish the performance requirements for the MSS Information Subsystem's Data Processing Assembly (DPA).

The DPA shall be capable of operating under the system limits as specified in paragraph 3.6.2.1.1 of this specification. The component computer programs which comprise the DPA shall operate without loss of data or degradation of time-critical functions under the specified system limits.

3.6.2.1.1 System Requirements. This paragraph establishes the limits and capacities that constrain the DPA design. The limits and capacities would be presented in tabular form (e.g., Table 3-1) and are grouped according to the following general categories:

Supervisory Programs	Support Programs
Application Programs	Data Base
Subsystems Operations	Test and Validation Programs

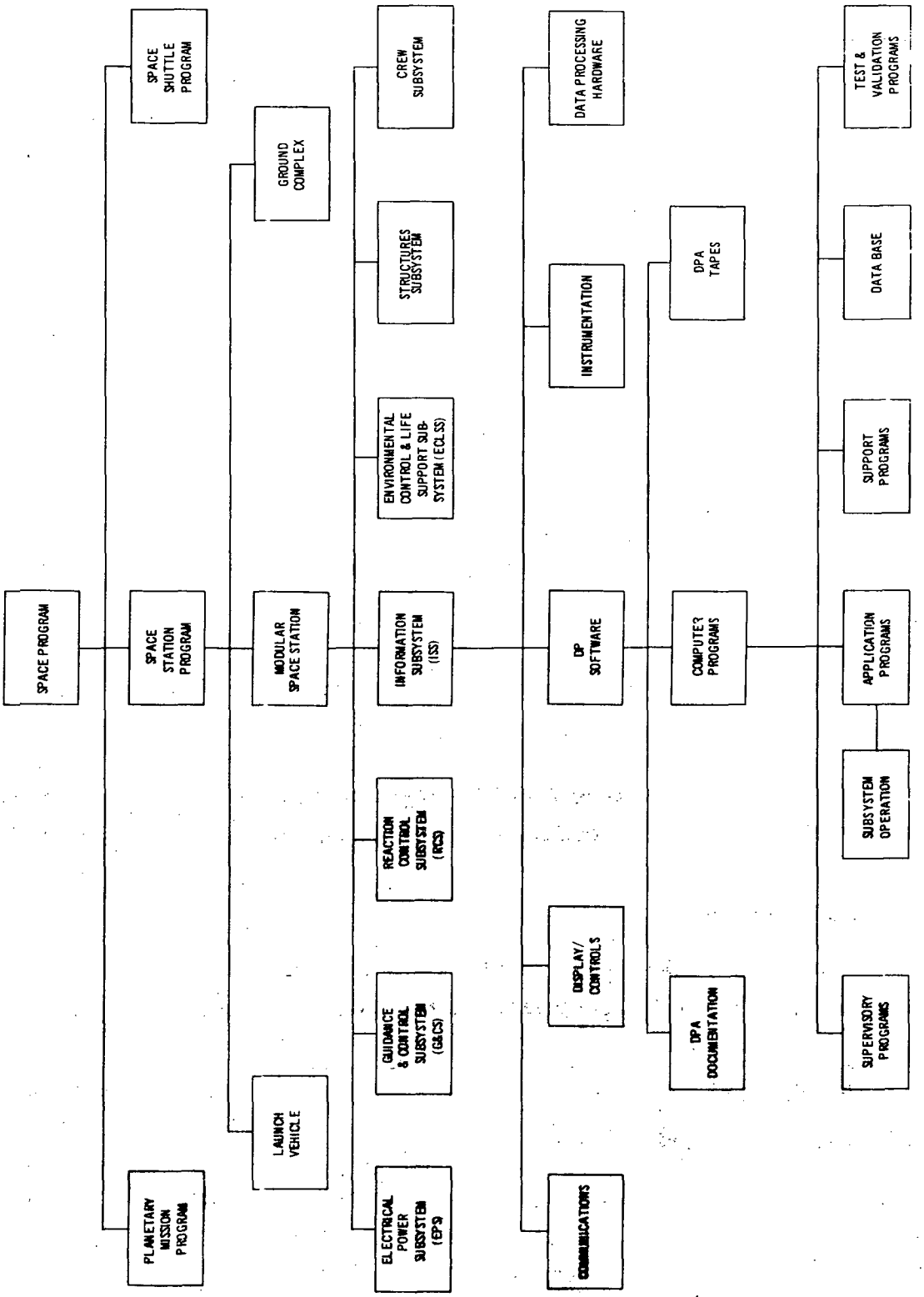


Figure 3-4. MAJOR COMPUTER PROGRAM COMPONENT HIERARCHY

Table 3-1. Supervisory Program System Limits and Capcities

Description	Units	Limits
Maximum number of relay satellite data link input/output channels	Channels	TBD
Maximum number of satellite communications destinations	Destimations	TBD
Maximum number of backup input/output channels	Sources	TBD
Maximum number of reporting sources	Sources	TBD
Maximum number of memory modules	Modules	TBD
Maximum number of display consoles	Consoles	TBD
Maximum number of addressable I/O devices	Devices	TBD
Maximum number of TBD	TBD	TBD

(The table above is given as an example. Further studies, falling outside the scope of this task, are required to determine the detailed limits and capacities for all categories of programs.)

Limits apply to both simulation and live requirements, unless either is specifically excluded.

3.6.2.1.2 Operational Requirements. The functional programs specified herein establish the operational requirements for this DPA.

Figure 3-5 illustrates the functional operation of the Data Processing Assembly, the relationship among DPA programs, and the relationship of these programs to other identified system operations.

3.6.2.1.2.1 Supervisory Programs. The supervisory programs shall provide the MSS with the capability to start, operate and supervise the other programs in the DPA; to schedule and control I/O operations; to control the timing and clocks in the DPA; to handle and control interrupts; to detect errors and error rates; to control, schedule and load programs from any of the program elements in the DPA; to provide multiprocessing; to allocate resources; to reconfigure the DPA; and to report to operators.

To fulfill these requirements, the Supervisory Programs shall be comprised of the following functions, which shall operate as specified in subsequent paragraphs:

a. Executive
b. I/O Scheduling and Control
c. Timing Control
d. Interrupt Handling

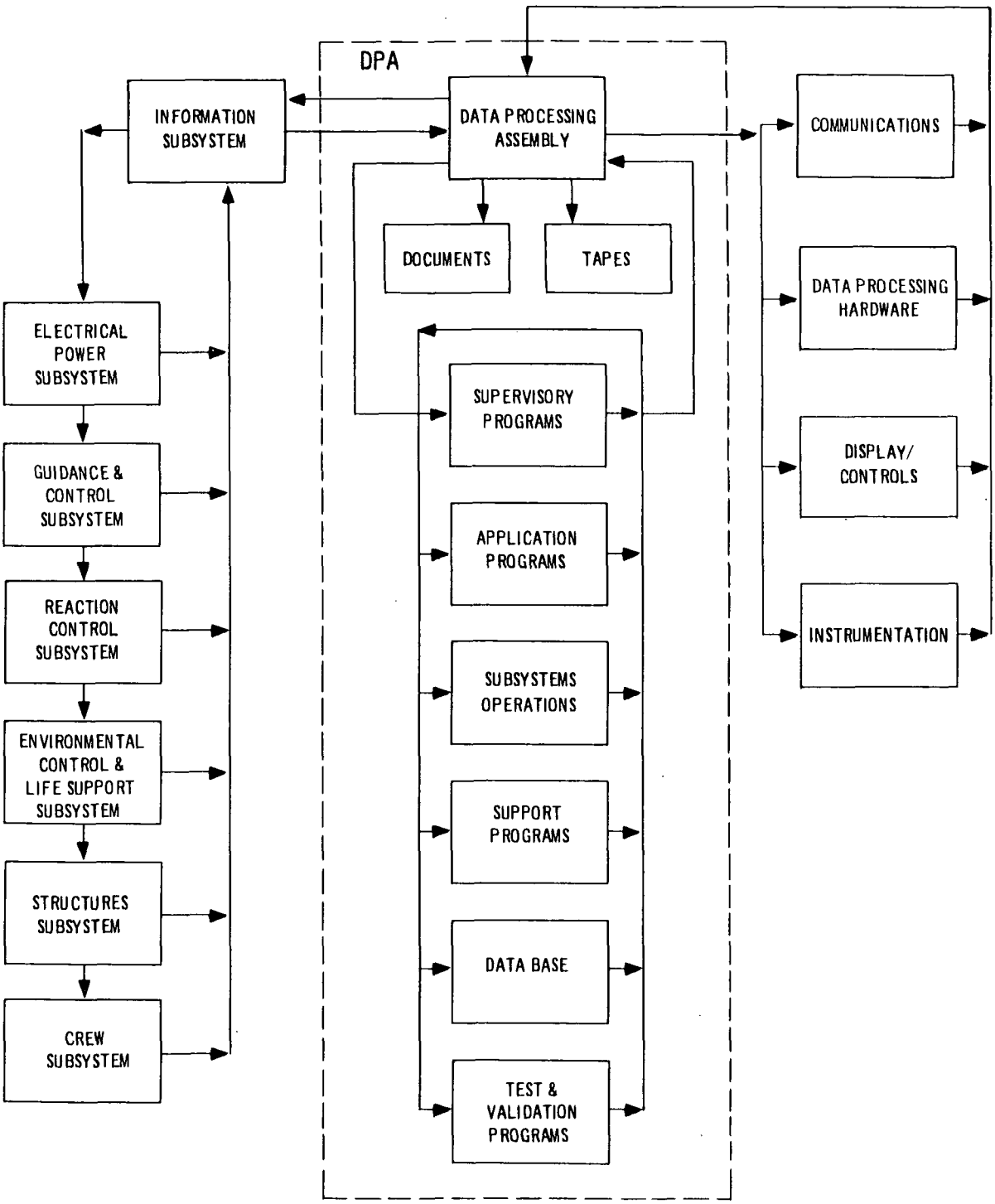


Figure 3-5. DPA LEVEL DIAGRAM

e. Program Control and Load
f. Task Scheduler
g. Multiprocessor Control
h. Resource Allocation

In addition the Executive shall have instant access to the On-Board Checkout and Control Program (see paragraph 3.6.2.1.2.2.5) and to the Generator/Starter Program (see paragraph 3.6.2.1.2.4.2) while the Display Formatting Program (see paragraph 3.6.2.1.2.4.3) will handle the reports to operators. The relationship between the supervisory programs is illustrated in Figure 3-6.

3.6.2.1.2.1.1 Executive. The Executive program shall provide the logic for the continuous operation of the DPA programs. It shall control and schedule the calling of other supervisory, application, support or data base programs or modules, and monitor the execution and interaction of these programs (see Figure 3-6).

3.6.2.1.2.1.2 I/O Scheduling and Control. The I/O Scheduling and Control program shall provide the logic for scheduling and controlling of all input and output operations and peripherals. The logic shall include the calling and operating of the following input/output modules:

a. I/O Operation Control
b. I/O Unit Control
c. I/O Channel Control

These modules shall provide the execution of the input/output requests at the proper times.

3.6.2.1.2.1.3 Timing Control. The Timing Control program shall provide the logic for controlling all system clocks and for synchronizing DPA operations with external vehicle clock time. In addition, it shall process requests for asynchronous actions based on time of day, initiate actions at predetermined clock time or after a given elapsed period of time. The logic shall include the calling and operating of the following timing control modules:

a. System Clock Control
b. Scheduler

3.6.2.1.2.1.4 Interrupt Handling. The Interrupt Handling Program shall provide the logic for analyzing the cause of the interrupt and take appropriate action by transferring control to one of its following subprograms:

a. Internal Interrupt Control
b. External Interrupt Control
c. Interrupt Status Control

Internal interrupts shall include interrupts for error or abnormal conditions internal to the CP such as parity errors.

External interrupts shall include interrupts for information transfer control, display and alarms control and manual data entry action control including switch actions.

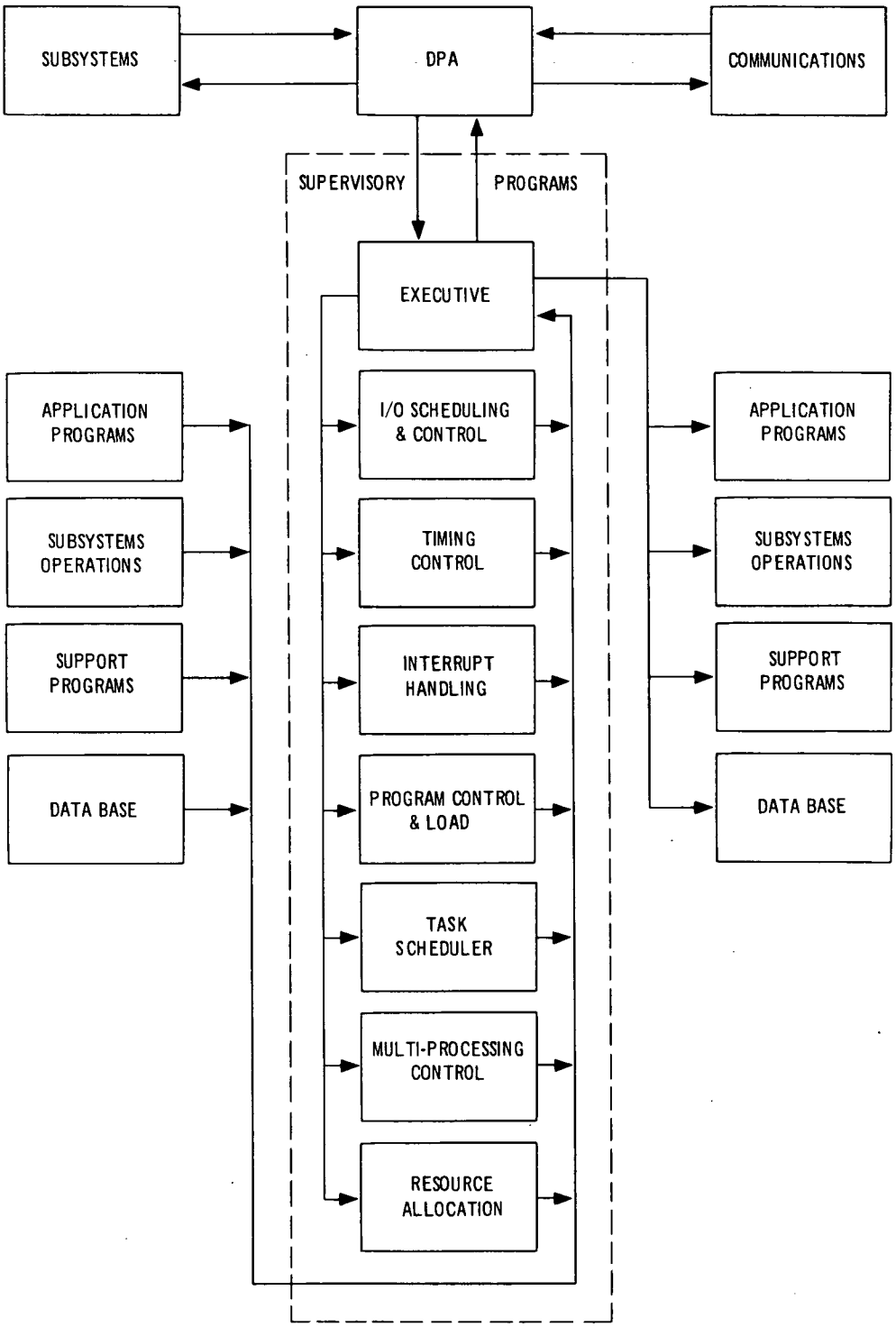


Figure 3.6. SUPERVISORY PROGRAMS FUNCTIONAL FLOW BLOCK DIAGRAM

3.6.2.1.2.1.5 Program Control and Load. The Program Control and Load program shall provide the logic for the calling, sequencing and loading of the application programs, supervisory programs, support programs and data base as necessary for the continuous operation of the MSS software.

For this purpose the logic shall include the calling, loading and operation of the following program control and load modules:

a. Program Module Control
b. Program Load Control
c. Data Base Maintenance

3.6.2.1.2.1.6 Task Scheduler. The Task Scheduler program shall provide the logic for scheduling the various tasks to be required, such as messages on a priority basis, building queues of items or requests, such as for I/O operations, scheduling machine functions when conflicts arise, such as in reconfigured operations, and scheduling algorithms to be processed. The logic shall include the calling and operating of the following task scheduler modules:

a. Algorithm Scheduling Processor
b. Queued Task Status Control

3.6.2.1.2.1.7 Multiprocessing Control. The Multiprocessing Control program shall provide the logic for enabling and controlling the processing of multiple functions simultaneously in more than one processor. The logic shall include the calling and execution of the following multiprocessing control subprograms:

a. Processor Control
b. Memory Control
c. Peripheral Control

3.6.2.1.2.1.8 Resource Allocation. The Resource Allocation program shall provide the logic for allocating hardware, peripherals, memory and storage in such a manner to obtain optimum system performance in response to changing data loadings, environmental conditions, reconfiguration, and consistent with the redundancy concept. The logic shall include the calling and execution of the following resource allocation subprograms:

a. Configuration Control
b. Storage Management
c. Fallback Control

3.6.2.1.2.2 Application Programs. The Application programs shall provide the MSS with the capability to perform its mission.

To fulfill these requirements, the Application programs shall be comprised of the following functions, which shall operate as specified in subsequent paragraphs:

a. Logistics Inventory Control
b. Command and Control
c. Experiment Data Management

d. Operational Data Management
e. On-board Checkout and Control
f. Subsystems Operations Management

The relationship between the Application programs and other components is illustrated in Figure 3-7.

3.6.2.1.2.2.1 Logistics Inventory Control. The Logistics Inventory Control program shall provide the logic for monitoring control of all logistics, the resupply long range projection and resupply status, the consumables trend detection, and usage prediction.

The logic shall include the calling and operation of the following

a. Resupply
b. Consumable Trend Detection
c. Consumable Usage Prediction

3.6.2.1.2.2.2 Command and Control. The Command and Control program shall provide the logic for:

Mission Activities Control
Orbital Operations Control
Message Generation Control

Mission Activities Control shall provide for the mission planning, mission scheduling and/or rescheduling, mission monitoring and evaluation. For this purpose the Mission Activities Control logic shall call and execute the following subprograms:

Mission Planning
Mission Scheduling/Rescheduling
Mission Monitoring
Real-Time Evaluation

Orbital Operations Control shall provide for the control of orbit, rendezvous, docking, and the evaluation of these functions. For this purpose the Orbital Operations Control shall call and execute the following subprograms:

Orbit Determination Control
Rendezvous Control
Docking Control
Performance Evaluation

Message Generation Control shall provide for command signal generation, message code formulating and verification, resolution of conflicting commands, and for message storage.

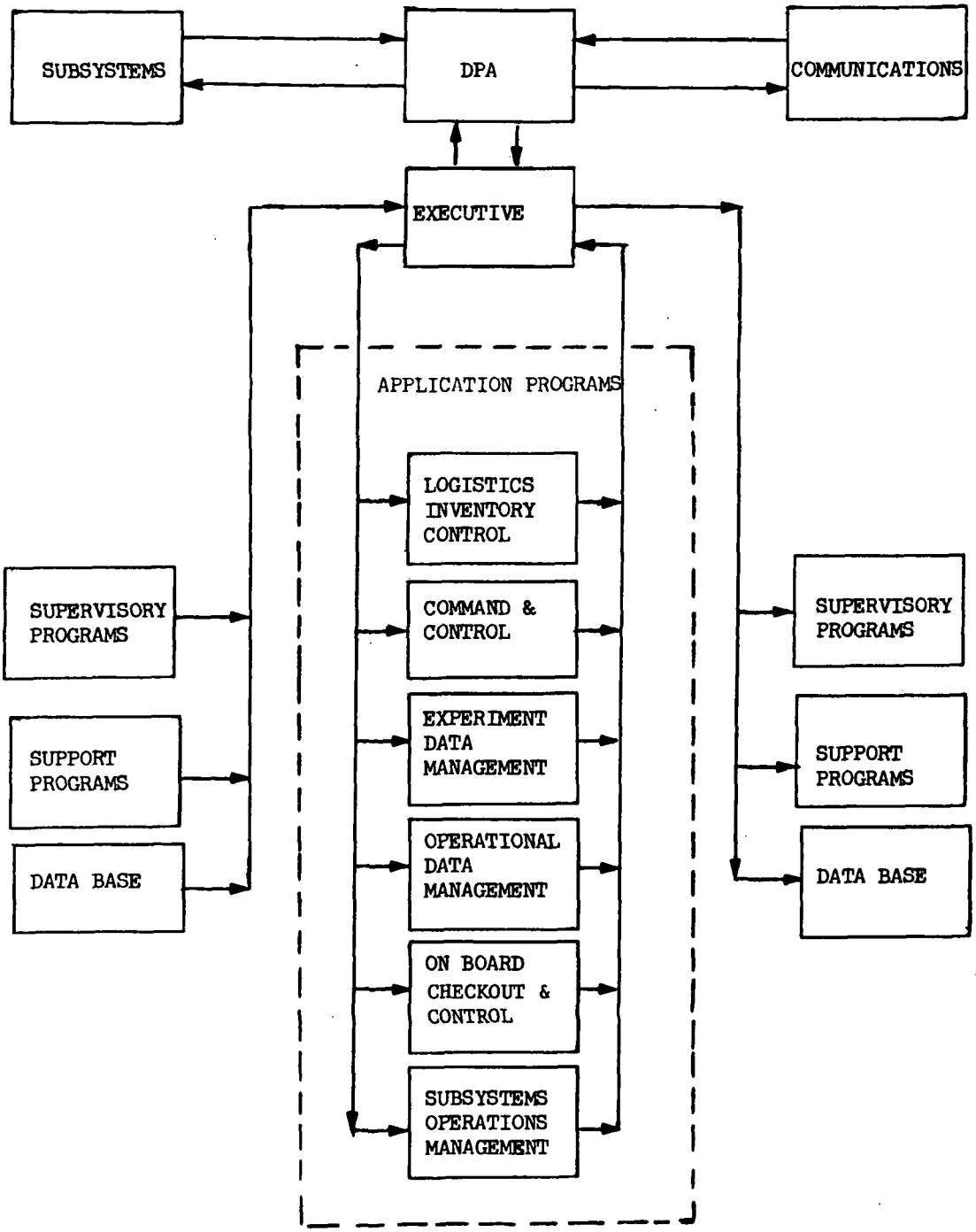


Figure 3-7. Application Programs Functional Flow Block Diagram

For this purpose the Message Generation Control shall call and execute the following subprograms:

Command Signal Generation
Command Conflict Resolution
Message Code Formulation
Message Code Verification
Message Storage

3.6.2.1.2.2.3 Experiment Data Management. The Experiment Data Management (EDM) program shall provide the logic for the handling and control of experiment data. For this purpose the program logic shall include the calling and operating of the following Experiment Data Management modules:

a. Real-Time Process Control
b. Data Storage and Retrieval
c. Data Compression
d. Data Analysis
e. Data Display

3.6.2.1.2.2.4 Operational Data Management. The Operational Data Management (ODM) program shall provide the logic for device control, shuttle alignment control, command execution and verification, and communications control. To fulfill these requirements, the ODM program logic shall include the calling and execution of the following ODM control subprograms:

a. Device Control
b. Shuttle/Module Alignment Control
c. Communications Control
d. Command Execution
e. Command Verification

Subsystems Operations Management is discussed in paragraph 3.6.2.1.2.3

3.6.2.1.2.2.5 On-Board Checkout and Control. The On-Board Checkout and Control (OBCO) program shall provide the logic for continuous monitoring for fault, malfunction, or breakdown in any of the subsystems on board the MSS and in case of malfunction take appropriate action. For this purpose the logic shall include transfer of control to any of the following subprograms:

a. Monitor Program
b. Fault Detection Program
c. Fault Isolation Program
d. Fault Prediction Program
e. Recertification Program
f. Calibration Program

3.6.2.1.2.3 Subsystems Operations Management. The Subsystems Operations Management program shall provide the logic for the data processing functions, including subsystem status control and interface, of the following subsystems:

Guidance and Control Subsystem (G&CS)
Electrical Power Subsystem (EPS)

Reaction Control Subsystem (RCS)
Environmental Control and Life Support Subsystem (ECLSS)
Crew Subsystem (CSS)
Structures Subsystem (SSS)

Figure 3-8 illustrates the functional data processing requirements for each of the subsystems. The following paragraphs describe the requirements for each of the subsystems.

3.6.2.1.2.3.1 Guidance and Control Subsystem. The Guidance and Control Subsystem program shall provide the logic for determination of the position and velocity of the MSS and the maneuvering requirements. To fulfill these requirements, the G&C subsystem program shall be comprised of the following subprograms which shall be called and executed:

a. Attitude Determination
b. Navigation Determination
c. Maneuver Determination
d. Control Moment Gyro Control
e. Reaction Control Subsystem Control

3.6.2.1.2.3.2 Electrical Power Subsystem. The Electrical Power Subsystem program shall provide the logic for storing, generating, regulating and controlling all electrical power required by the MSS. To fulfill these requirements the Electrical Power Subsystem program shall be comprised of the following subprograms:

a. Solar Array Pointing and Control
b. Inverter Control
c. Fuel Cell Control
d. Central Power Control
e. Current Flow Control

3.6.2.1.2.3.3 Reaction Control Subsystem. The Reaction Control Subsystem program shall provide the logic for control of the forces required to stabilize, maintain momentum of and maneuver the MSS. To fulfill these requirements the RCS program shall call and execute the following subprograms:

a. Thrust Valve Control
b. Nitrogent Quantity Balance Control

3.6.2.1.2.3.4 Environmental Control and Life Support Subsystem. The Environmental Control and Life Support Subsystem program shall provide the logic for control of a habitable environment for crew and equipment. For this purpose the following subprograms shall be available for execution by the ECLSS program:

a. Gaseous Storage Control
b. CO_2 Management
c. Atmospheric Control
d. Thermal Control
e. Water Management
f. Food Management
g. Special Life Support

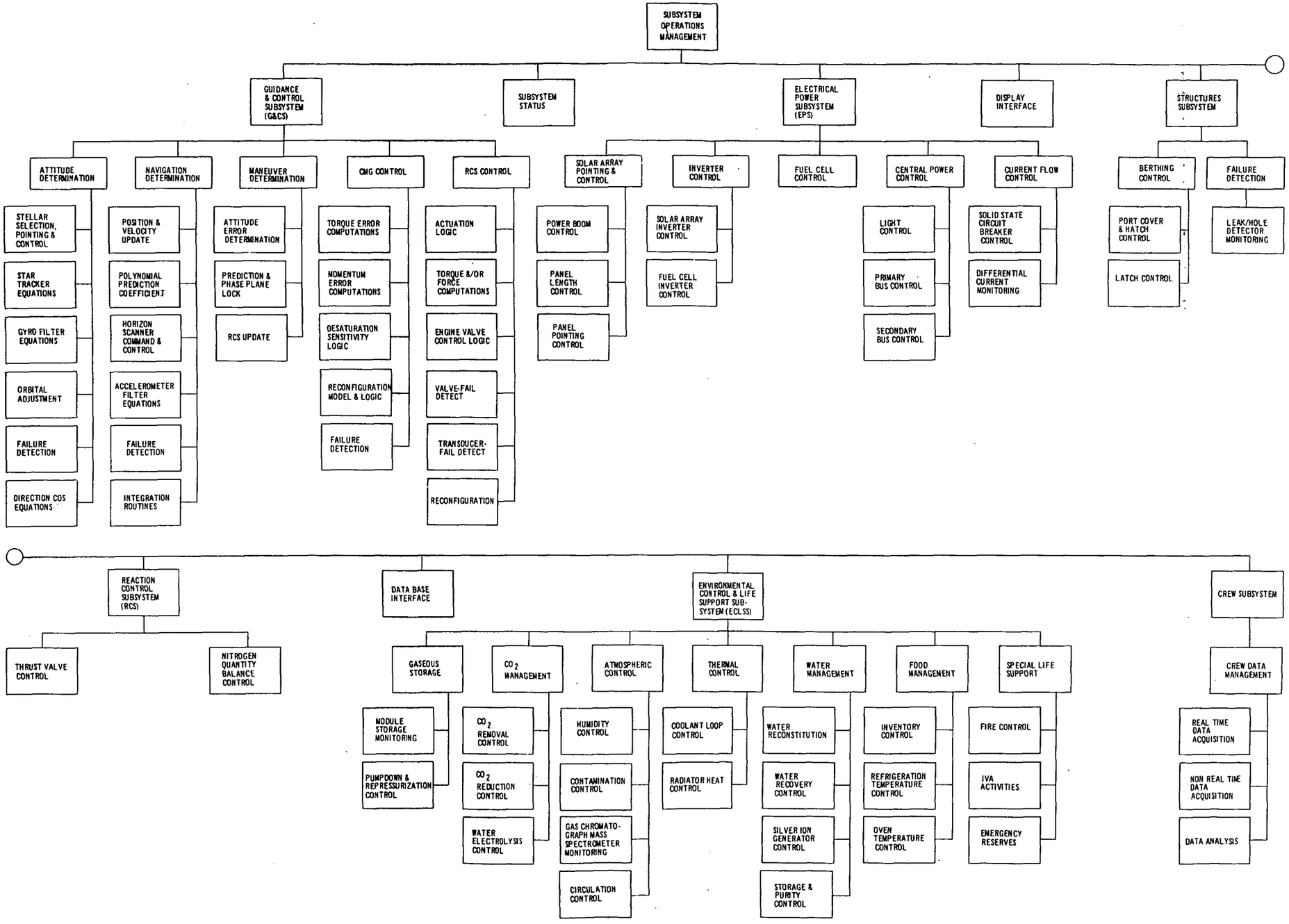


Figure 3-8. Functional Data Processing Requirements of Subsystems Operations

3.6.2.1.2.3.5 Structures Subsystem. The Structures Subsystem (SSS) program shall provide the logic for berthing and structural failure detection. For this purpose the SSS control program shall call and execute the following subprograms:

a. Berthing Control
b. Failure Detection

The Berthing Control subprogram shall access the Port Cover and Hatch Control module, and the Latch Control module to execute its functions, while the Failure Detection subprogram shall access the Leak/Hole Detector Monitoring module.

3.6.2.1.2.3.6 Crew Subsystem. The Crew Subsystem program shall provide the logic for medical and personal data management for the crew. For this purpose the Crew Data Management logic shall call and execute the following modules:

a. Real-Time Data Acquisition
b. Non-Real-Time Data Acquisition
c. Data Analysis

3.6.2.1.2.4 Support Programs. The Support Programs shall provide the MSS with the capability to develop, test and maintain the operational system.

To fulfill these requirements, the Support Programs shall be comprised of the following categories of functional programs, which shall operate as specified in subsequent paragraphs:

a. Simultaion Programs
b. Utility Programs
C. Application Support Programs

The relationship between the Support Programs and the other programs in the DPA is illustrated in Figure 3-9.

3.6.2.1.2.4.1 Simulation Programs. The Simulation Programs shall provide the capability to generate and use artificially generated information for the purpose of:

a. Testing all components of the data processing assembly
b. Training personnel

To fulfill these requirements, the Simulation Programs shall be comprised of the following functional subprograms:

a. Data Generation
b. Environment Generation and Maintenance
c. Operator Display Generation
d. Operator Control Interpretation and Generation
e. Application Program Simulation
f. Supervisory Simulation Programs

The Simulation Program Tree is illustrated in Figure 3-10.

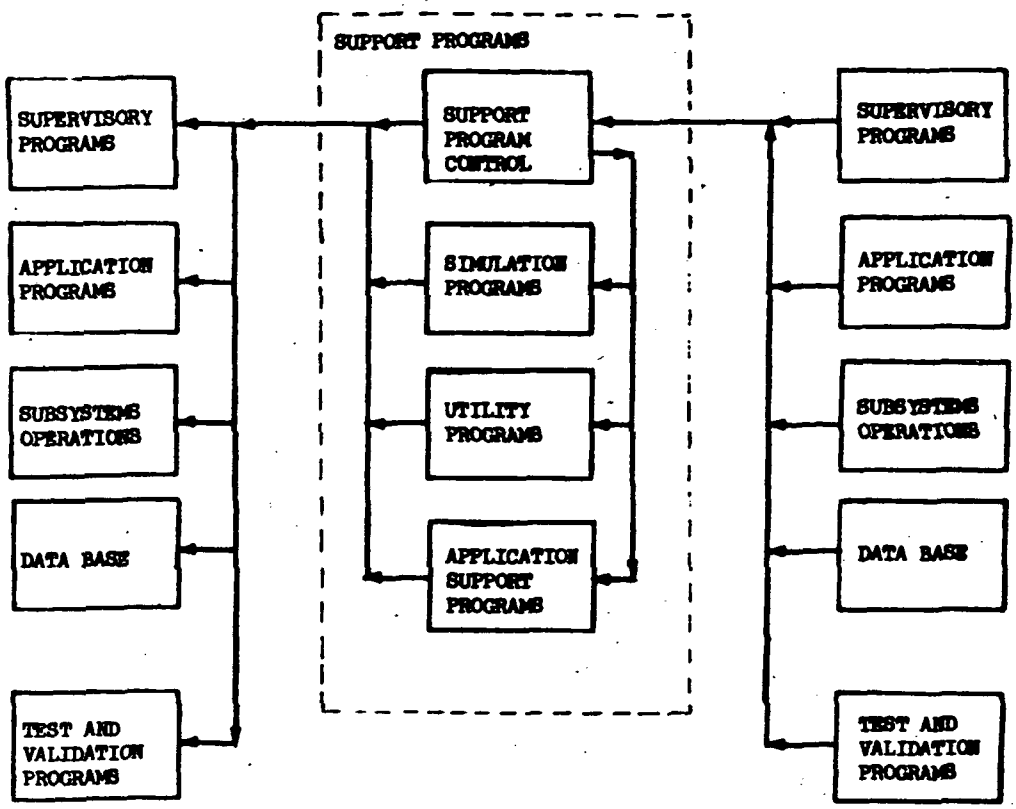


Figure 3-9. Support Programs Functional Block Diagram

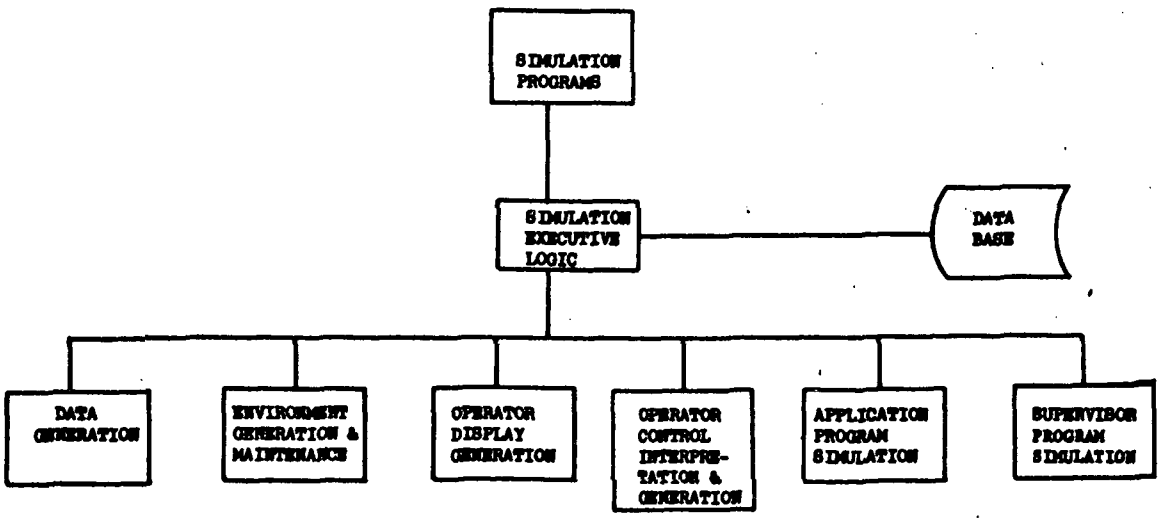


Figure 3-10. Simulation Program Tree

3.6.2.1.2.4.2 Utility Programs. The Utility Programs shall provide the logic for translating higher-order language programs, modules, routines, and data into binary machine language, generating dictionaries for program use, generate master tapes and listings, start and restart the system, maintain files on tape, disk or drum, record selected parts of memory, and symbolically change or correct existing programs.

To fulfill these requirements, the Utility Program logic shall include the calling and execution of the following subprograms:

a. Compiler
b. Assembler
c. CP Symbolic Modification Program
d. Master Tape Generation Program
e. Recording Program
f. Storage Dumping Program
g. Generator/Start Program

The Utility Program Tree is illustrated in Figure 3-11.

The following paragraphs briefly describe each of the above-mentioned subprograms.

3.6.2.1.2.4.2.1 Compiler. Compiler shall provide the capability to compile computer programs, routines and modules scripted in the TBD Higher-Order Language. Compilation shall provide the capability to generate relocatable and reentrant object coded computer program components. Computer program compilation shall not require manual intervention; the compiler shall process, without interruption, a sequence of computer program components (batched processing) and shall be capable of compiling itself.

To fulfill its requirements, the compilation shall be divided into the following functions:

Process Primary Control Inputs
Obtain Source-Language Inputs
Generate Source-Language Level Outputs
Decode Source-Language Computer Program
Generate Object- Computer Program
Generate Object-Level Output
Produce Debug-Aid Listings
Check for External and Internal Error

3.6.2.1.2.4.2.2 Assembler. Assembler shall provide the capability to assemble data expressed as symbolic inputs. These data shall be assembled into binary on a one-to-one basis, and written onto a magnetic tape or assembled into a binary form suitable for display.

To fulfill its requirements, assembler shall be divided into the following functions:

Process Control Inputs
Obtain and Assemble Data

Generate Binary Data
Generate Listing
Generate Binary Tape

3.6.2.1.2.4.2.3 CP Symbolic Modification Program. The CP Symbolic Modification Program shall provide the capability to modify the binary content of a computer program component as directed by symbolic assembly language input.

To fulfill its requirements, the CP Symbolic Modification Program shall be divided into the following functions:

Process Control Inputs
Obtain and Process Symbolic Modification Declarations
Process Symbolic Processing Statements
Generate Requested Listings
Check for Internal and External Errors

3.6.2.1.2.4.2.4 Master Tape Generation Program. The Master Tape Generation Program shall provide the capability to generate the magnetic binary master tapes for DPA.

The exact format of the binary master tape within the constraints of Master Tape Generation Program shall be determined by user-specified inputs. It shall provide the capability to list the new master tape and to update and list the inventory of modifications loaded on the master tape.

To fulfill its requirements, the Master Tape Generation Program shall be divided into the following functions:

Process Control Data
Generate Master Tape
Catalog Master Tape
List Inventory
Compare Master Tapes

3.6.2.1.2.4.2.5 Recording Program. The Recording Program shall provide the capability to write data from selected memory storage areas onto magnetic tape during the operation of any of the DPA programs. It shall have the capability to record data, based on selections made by the user before the operation of the program. To fulfill its requirements, Recording shall be divided into the following functions:

Process Control Data
Generate a Recording Tape Record Image
Select a Recording Tape Unit
Generate a Recording Tape

3.6.2.1.2.4.2.6 Storage Dumping Program. The Storage Dumping Program shall provide the capability to print the contents of memory, disks or magnetic tapes in TBD formats.

To fulfill its requirements, Storage Dumping shall be divided into the following functions:

Process Control Data
Dump Storage Data

3.6.2.1.2.4.2.7 Generator/Start Program. The Generator/Start Program shall provide the capability to start and restart the MSS Master Program and put Executive in control and generate the associated necessary data. In addition, it shall provide the capability to start any of the other MSS related program systems.

To fulfill its requirements, Generator/Start shall be divided into the following functions:

Start/Restart
Generate Dictionary
Generate Data Base
Generate Own File Library

3.6.2.1.2.4.3 Application Support Programs. The Application Support Programs shall provide the capability and logic to maintain files and records, and do general bookkeeping functions fo facilitate the operations of the main application programs, and provide a data recording and data reduction capability. This logic shall include the calling and execution of the following support modules:

a. Personnel Records Maintenance
b. Catalog Maintenance
c. Data Base Verification and Maintenance
d. Information Retrieval
e. Display Formatting
f. Telemetry Formatting
g. Library Routines
h. Data Recording Program
i. Data Reduction Program

The Application Support Programs Tree is illustrated in Figure 3-12.

3.6.2.1.2.5 Test and Validation Program. The Test and Validation Programs shall provide the logic for a comprehensive and flexible test driver capable of initializing data base, simulating interfaces and flight hardware, specifying inputs, and using supervisory and applications programs for simulation.

Its logic shall also include reduction of test data, diagnostic and debugging aids and a storage dump capability.

To fulfill its requirements, the Test and Validation Program shall be comprised of the following functions:

Flight Hardware Test Support
Test Data Reduction
Supervisory Program Simulation

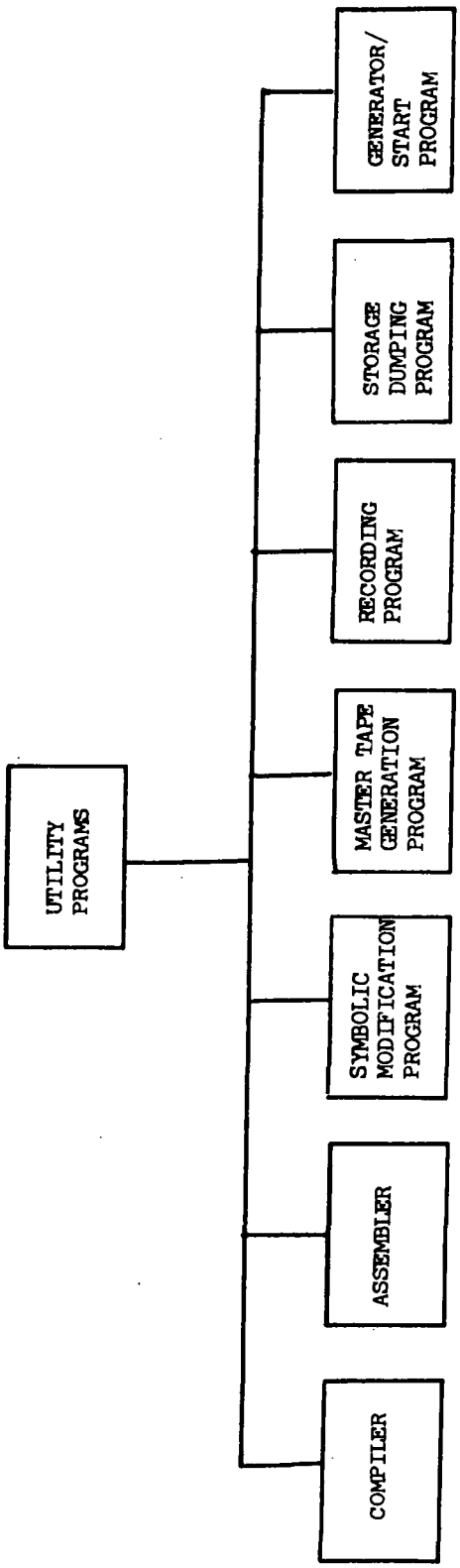


Figure 3-11. Utility Programs Tree

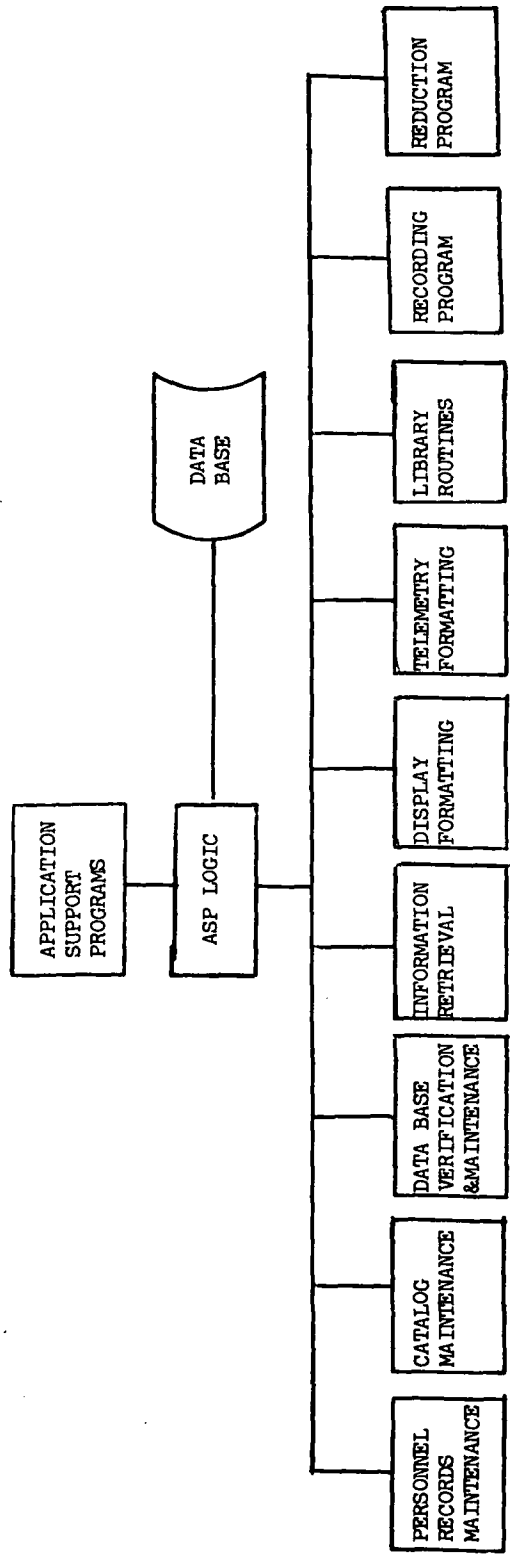


Figure 3-12. Application Support Programs Tree

Application Program Simulation
Diagnostic and Debugging Programs
Storage Dump

The Supervisory and Application Program Simulation functions shall be fulfilled by the like named subprograms described in paragraph 3.6.2.1.2.4.1. The Test Data Reduction function shall be fulfilled by the Data Reduction Program indicated in paragraph 3.6.2.1.2.4.3. The Storage Dump function shall be fulfilled by the Storage Dumping Program described in paragraph 3.6.2.1.2.4.2.6.

The Test and Validation Program Tree is illustrated in Figure 3-13.

3.6.2.1.3 Data Base Requirements. The DPA shall incorporate and utilize the data base items specified herein. These data are grouped into categories as follows:

a. Operational Reference Data

1. Star Charts
2. Ephemerides
3. Coordinate Origins
4. Program Constants
5. Initial Conditions
6. TBD

b. Subsystems Data

1. Positional Data
2. Facility Data
3. Navigational Data
4. Propellant Status Data
5. TBD

c. TBD

Definition, nomenclature, dimensions, units and location of data in the data base is TBD.

3.6.2.1.4 Human Performance. The following subsections establish the human performance/human engineering requirements by which computer program initiated/controlled displays and switch actions specified in this specification are designed. Human performance requirements assure that an effective interface exists between the crew and display consoles. The human operator when interfacing with the display console, performs data sensing, information processing, and responding functions. Errors, time delays and, in extreme situations, disruptions can occur in each of these functions by overstressing or exceeding man's capabilities, or forcing man to perform additional subfunctions which are themselves subject to errors and time delays. The human performance requirements are stated with the cognizance of these limiting characteristics.

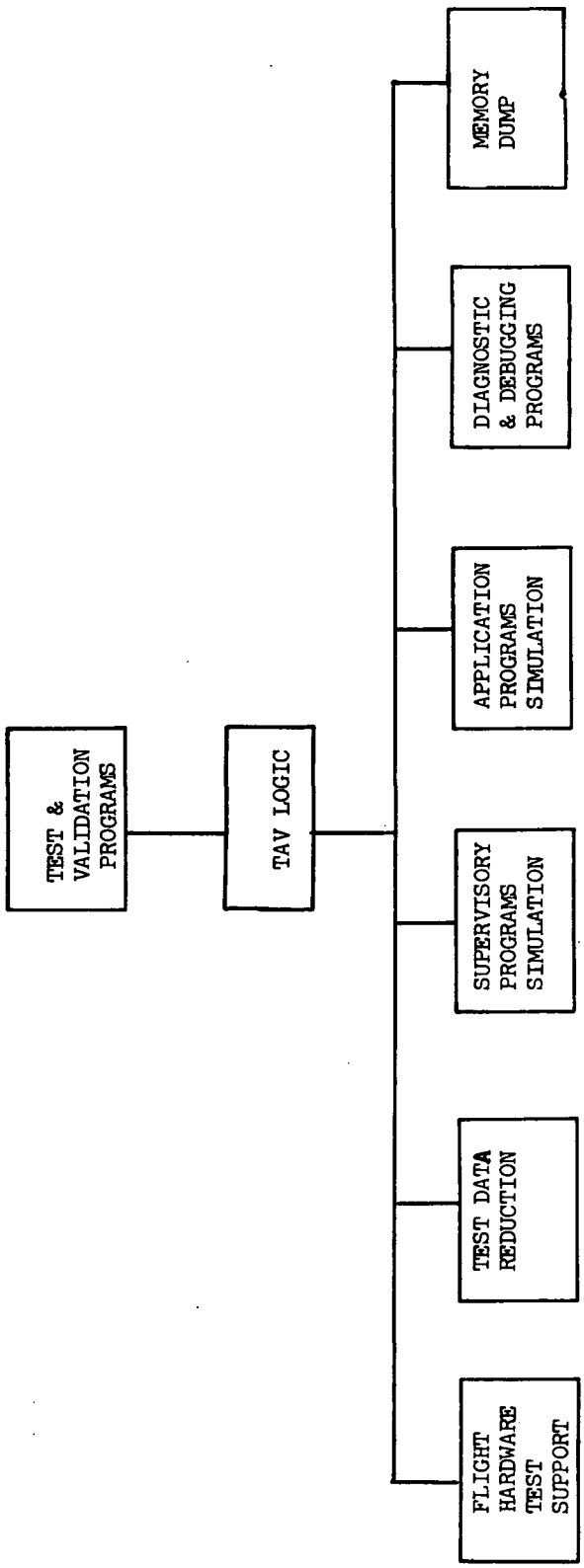


Figure 3-13. Test and Validation Program Tree

The goal of the human performance requirements is to specify designs which enable the crew to acquire and maintain levels of performance sufficient to meet the system requirements. The objectives of the human performance requirements are to:

a. Reduce errors in the perception of information

b. Ensure the capability of rapid perception of information

c. Reduce errors in information processing by the operator

d. Ensure the capability of rapid assessment and decision making

e. Reduce errors in the taking of switch actions

f. Ensure the capability of accurate and timely switch action responses

g. Ensure timely direction of operator attention to situations requiring intervention

3.6.2.1.4.1 Displays. All Display format designs for MSS are described in the MSS Phase B Preliminary Design Report, SD 71-217-4.

Data which are transposed or transformed by the computer into units to be used by the operator reduces operator errors, decreases action times, and permits more efficient operation.

Alarm displays shall be utilized to denote to the operator conditions which require immediate attention or rapid remedial action. (The types of available alarms are blinking symbols, audible alarms, anf forced unique symbols.)

3.6.2.1.4.2 Feedback. Every operation action shall be followed by a positive indication of program acceptance or rejection. Feedback on data entry and request switch actions reduces needless repetition of switch actions by informing the operator of the reasons for delays in switch action processing.

3.6.2.1.4.3 Timing Considerations. To effectively perform in an automated environment, an operator requires rapid responses to requests or inputs to the system and rapid access to alarm conditions.

The DPA shall incorporate a design which will service the requirements of the operator in a timely manner to meet immediate operational needs. The specific display information will depend on system load.

3.6.2.2 Interface Requirements

The DPA CEIs interrelate in supporting the MSS mission. The relation of the CEIs, including equipment and interfacing subsystems, is defined in the following subsections.

The interface functional block diagrams and interface definitions define the interfacing relationships.

Figure 3-14 illustrates the DPA Functional Interfacing Block Diagram.

3.6.2.2.1 Ground Complex Interface. DPA interfaces through the Communications Subsystem with the Ground Complex Program Element through the latter's Mission Support subsystem for the prelaunch checkout and launch (Launch Control Facility), for tracking and orbital guidance (Mission Support Facility), and rendezvous/docking support.

The Computer Program System Interface Specification shall specify and define the details for this interface.

3.6.2.2.2 Remote Space Module/Space Shuttle Interface. DPA interfaces with Remote Space modules and Space Shuttle are TBD. The Computer Program System Interface Specification shall specify and define in detail these interface requirements.

3.6.2.2.3 Subsystems Interface. The DPA interface requirements shall be specified and defined in detail in the Subsystem Requirements Specifications for the following subsystems:

Electrical Power Subsystem
Reaction Control Subsystem
Guidance and Control Subsystem
Environmental Control and Life Support Subsystem
Structures Subsystem
Crew Subsystem
Communication Subsystem

3.6.2.2.4 Computer Program/Equipment Interface. The functional relationship of DPA to interfacing equipment is depicted in Figure 3-15 DPA/Equipment Interface. Details on the Computer/Control Processing Units, mass memory, and peripheral equipment are TBD.

3.6.2.3 Design Requirements

The design of the DPA with regard to the system, operational, and data base performance requirements specified in paragraphs 3.6.2.1.1, 3.6.2.1.2, and 3.6.2.1.3 respectively, shall be accomplished in accordance with the program design requirements in the following paragraphs.

3.6.2.3.1 Programming Language. The DPA computer programs shall be coded in the TBD higher-order programming language.

3.6.2.3.2 Program Design Standards. The computer program standards as described in the Modular Space Station Computer Program Standards and Conventions, shall be adhered to.

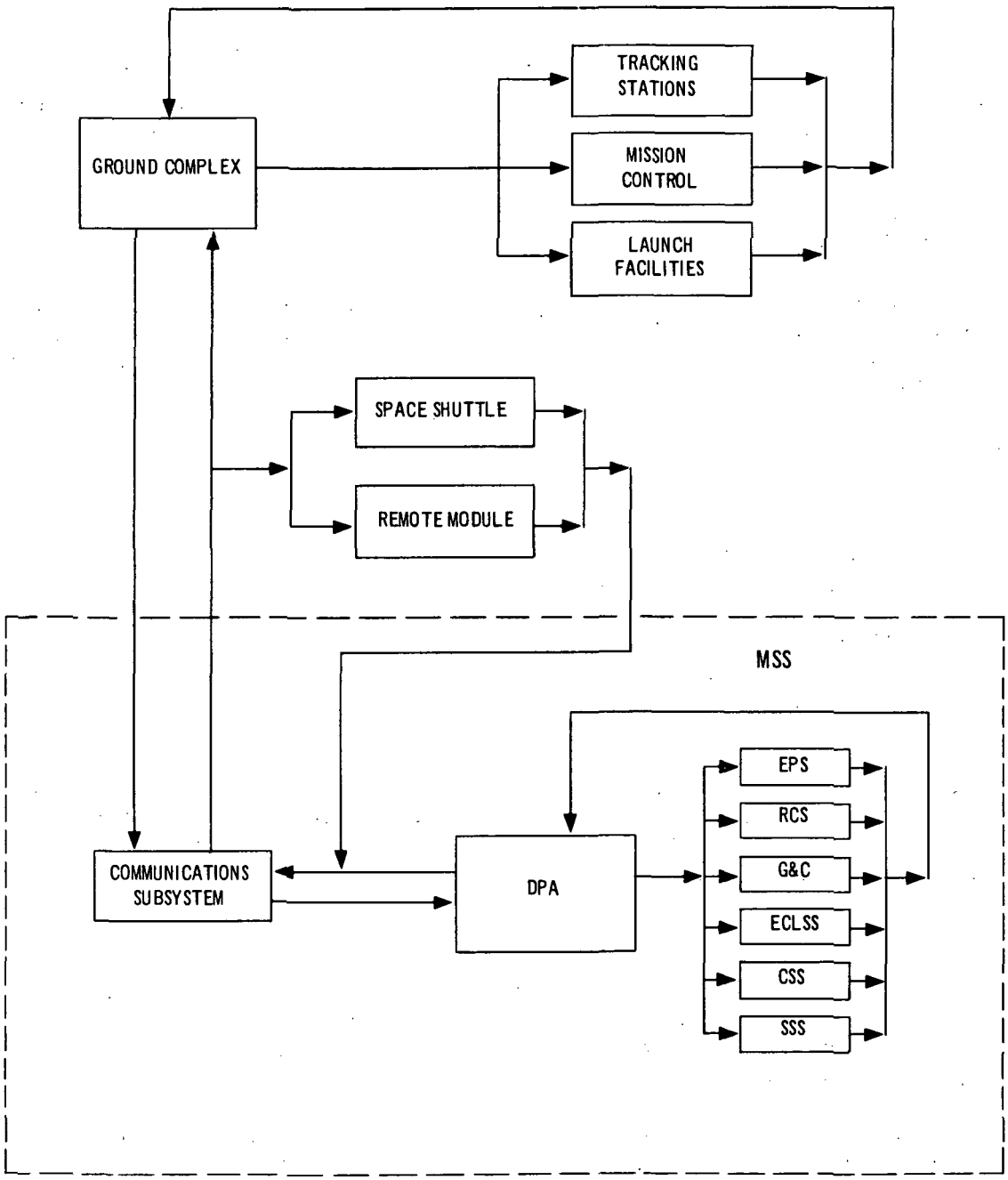


Figure 3-14. DPA INTERFACE BLOCK DIAGRAM

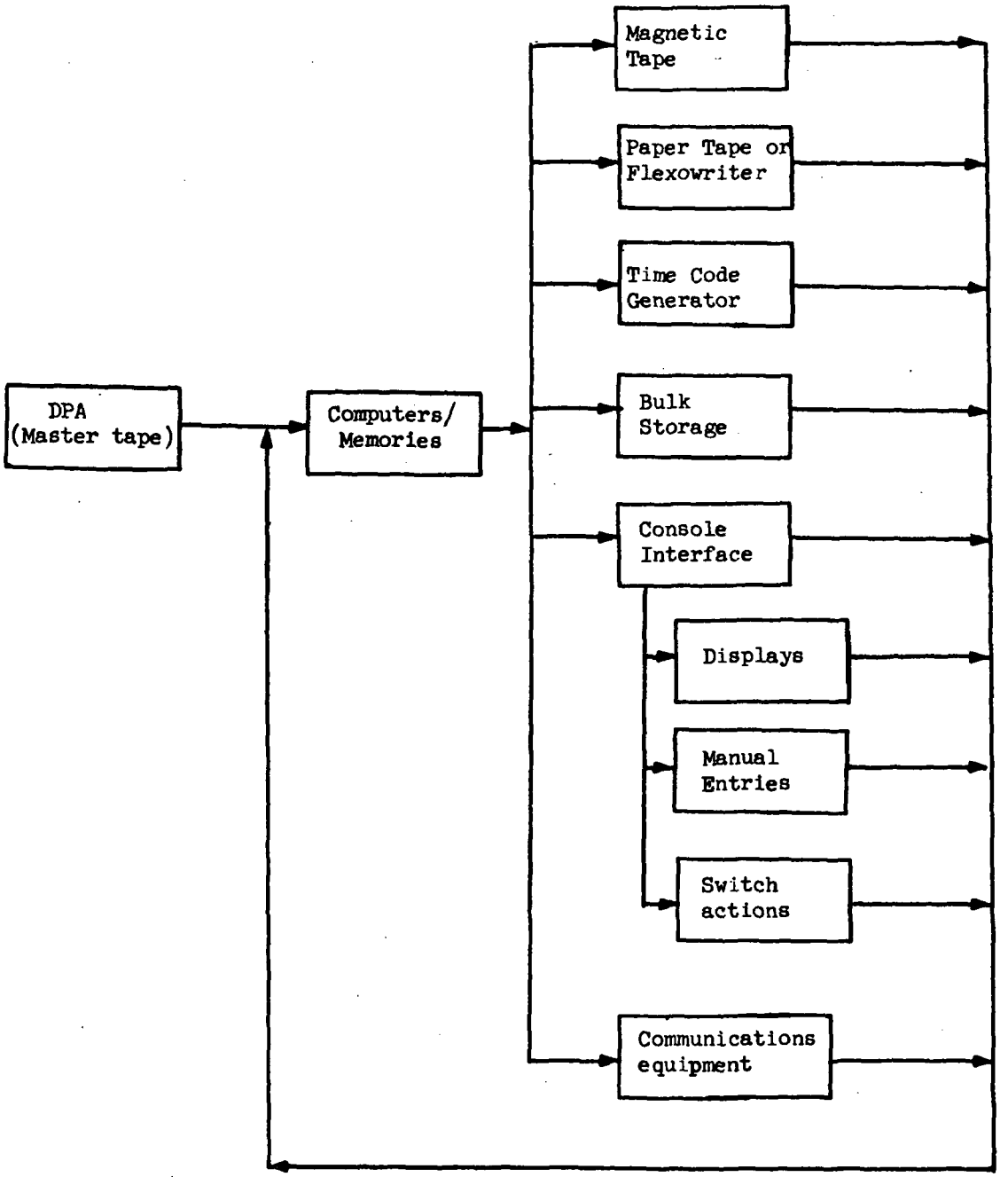


Figure 3-15. DPA/Equipment Interface Block Diagram

3.6.2.3.3 Program Coding Conventions. The Computer Program Coding Conventions as described in the Modular Space Station Computer Program Standards and Conventions shall be adhered to.

3.6.2.3.4 Test Features for the Executive Control Program. The Executive Control Program shall have the capability to collect and maintain data for evaluation of its performance. The data shall include information relating to the following:

a. The timlines with which the supervisory programs, application programs, and support programs are operated in relation to their data

b. The efficiency of CPU utilization

c. Abnormal events, such as data loss or serious functional delays

3.6.2.3.5 Expandability. Provision shall be made for growth capacity, to permit expansion and modification of the system. This requirement shall be met by allowing for a minimum of TBD percent spares for computer instructions and data storage, by modularizing program components, and by generalizing program design to minimize the impact of subsequent changes in required interfaces and to permit the incorporation of diverse future capabilities.

3.6.2.4 Data Rates

TBD

3.6.2.5 Timing Requirements

TBD

3.6.3 QUALITY ASSURANCE PROVISIONS

The purpose of the provisions contained in this section are:

a. To specify the requirements and conditions for formal verification of the performance and design requirements specified in Section 2.

b. To define requirements for equipment, facilities, and computer programs to support contractor design, development, and testing of the DPA programs.

c. To specify the Category I test methodology to be employed in formally verifying that the DPA programs satisfy all performance and design requirements specified in Section 3.6.7.

d. To provide accountability of each Section 3.6.2 requirement and the applicable test methods by means of a verification cross reference index. Qualification requirements associated with acceptance of the DPA shall be satisfied by the completion of the Quality Assurance Program specified herein.

This quality assurance program provides for continuous monitoring of the quality of the DPA programs throughout the entire design, development and production process. Particular emphasis and control shall be given the Computer Programming Test and Evaluation phase specified below. Selected data from this activity shall be retained to be used as required to verify satisfaction of requirements not covered during preliminary and formal qualification testing.

Formal quality assurance of the DPA programs shall be limited to the provisions of this section.

3.6.3.1 Category I Test

Category I testing shall be conducted on the DPA programs to verify compliance with the performance and design requirements specified in Section 3.6.2. This Category I testing includes all testing of the DPA programs. Category I testing shall be divided into three types: Computer Programming Test and Evaluation (CPT&E), Preliminary Qualification Tests (PQT), and Formal Qualification Tests (FQT).

The primary source of data to verify satisfaction of the performance and design requirements at the subfunction level and below shall be CPT&E. These data shall be retained and made available during PQT and FQT as required to support the verification of the higher level requirements. Since CPT&E is conducted while the DPA is still under development and change, the contractor shall clearly establish that the supportive data are relevant to the particular performance and design requirements subsumed under the subfunction or higher level requirements being examined in PQT and FQT. These CPT&E data shall, in addition, be the data on the set or subset of Computer Program Components used in the PQT or FQT being conducted. PQT shall be used to demonstrate satisfactory progress in the design and development of the CPCEI and satisfaction of requirements specified herein. Formal Qualification Tests shall be conducted on the integrated DPA programs utilizing DPA level inputs and outputs and shall verify the performance and design requirements at the DPA level.

Detailed requirements pertaining to CPT&E, PQT, and FQT are contained in the following subparagraphs.

3.6.3.1.1 Computer Programming Test And Evaluation. As part of the design and development of the CPCEIs, testing shall be conducted prior to and in parallel with preliminary and formal qualification testing. This testing, designated Computer Programming Test and Evaluation (CPT&E), shall be designed to provide an on-going program of control and evaluation of product quality throughout the entire design and development process and shall constitute, together with preliminary and formal qualification testing, the quality assurance program for the CPCEI.

The computer programming subcontractor shall develop and internally document this CPT&E program. CPT&E shall be conducted by the computer programming subcontractor at his design and development facilities. Descriptions and results from the subsets of tests which demonstrate satisfaction of the CPCEI Part I specifications will be made available as required to support qualification testing of portions of the CPCEI for those cases in which specific, detailed data is

required. Such data shall consist of test records, input listings, data reduction printouts of recording made during tests, and analysis of observed results and of reduced data. The testing described above will provide a systematic verification of the detailed performance requirements which is further described in the computer programming subcontractor's Development Specification and Validation Test Plans. There are no qualification requirements identified explicitly in this document for CPT&E.

3.6.3.1.2 Preliminary Qualification Tests. The scope of preliminary qualification testing of the DPA shall be limited to demonstrating that developed portions of the DPA satisfy the specific requirements of Section 3.6.2 to be determined at a later date. These requirements shall be grouped into logical sets of tests designed to demonstrate to NASA/MSC that progress is being accomplished in the development of the total DPA. This progress shall be demonstrated by scheduling preliminary qualification tests during design and development and by demonstrating increased complexity and total production of the DPA programs over time.

Successful completion of PQT will demonstrate progress toward formal qualification of the DPA. Descriptions and results from those subsets of CPT&E tests which demonstrate satisfaction of the DPA Part I specifications shall be retained along with results of PQT and shall be made available as required to support formal qualification testing of portions of the DPA for those cases in which specific, detailed data is required during FQT. Such data shall consist of test records, input listings, data reduction printouts of recording made during the tests, and analyses of observed results and of reduced data.

Preliminary qualification testing shall be limited to the requirements and associated methods to be enumerated in this subparagraph at a later date. The minimum requirements for conducting the PQT phase of the quality assurance program shall also be listed.

3.6.3.1.3 Formal Qualification Tests. Formal qualification testing of the DPA shall be conducted under the provisions of this subparagraph to verify that all performance and design requirements specified in Section 3.6.2 are met. Formal qualification testing will demonstrate satisfaction of the verification requirements by the methods specified herein. This testing shall be conducted at the DPA level and verification that the DPA programs satisfy the performance and design requirements together with supporting data from CPT&E and PQT as required shall constitute satisfaction of all requirements in Section 3.6.2. The methods of verification to be specified in this subparagraph shall include Inspection, Analysis, Demonstration Tests, Review of Prior Data. A table shall provide a cross-reference index showing classification of testing, methods of verification, and paragraph number for each performance and design requirements in Section 3.6.2.

3.6.3.2 Category II System Test

TBD

3.6.4 COMMONALITY OF FUNCTIONAL REQUIREMENTS

The requirements established by this top level specificationnfor the DPA have been optimized based upon the requirements for the Modular Space Station. However, it should be noted that in certain areas, the Part I Specifications for the MSS and other remote space modules are functionally equivalent. This implies that in such equivalent areas the requirements of this specification and the requirements for the other space modules can be satisfied by, result in, or imply the use of a common programming logic.

3.6.5 NOTES

3.6.6 APPLICABLE DOCUMENTS

TBD

3.6.7 APPENDIXES

3.6.7.1 Major Contract End Item (CEI) Matrix

This matrix, Table 3-2, indicates the requirement for a CEI for each of the computer programs, subprograms, routines, modules, discussed in this system specification.

Table 3-2. Major Contract End Item Matrix

Computer/Program Module \ Major CEI	Subsystem Requirements Specification	Software Operational Design Specification	Comp. Progr. Development Specification Part I	Comp. Progr. Detail Specification Part II	Comp. Progr. Module Listing	Test Plans & Requirements Specification	Handbook	Users Manual
Supervisory Programs			X			X		
I/O Scheduling & Control				X				
I/O Operation Control					X			
I/O Unit Control					X			
I/O Channel Control					X			
Timing Control & Scheduling				X				
System Clock Control					X			
Scheduler					X			
Algorithm Sched. Processor					X			
Queued Task Status Control					X			
Program Control & Load				X				
Program Module Control					X			
Program Load Control					X			
Data Base Maintenance					X			
Interrupt Handling				X				
Internal Interrupt Control					X			
External Interrupt Control					X			
Interrupt Status Control					X			
Multi Processing Control				X				
Processor Control					X			
Memory Control					X			
Peripheral Control					X			
Resource Allocation				X				
Configuration Control Cataloger					X			
Storage Management Allocator					X			
Fallback Control Switchover					X			
Executive				X	X	X		

X indicates CEI required.

Table 3-2. Major Contract End Item Matrix (cont.)

Computer/Program Module \ Major CEI	Subsystem Requirements Specification	Software Operational Design Specification	Comp. Prog. Development Specification Part I	Comp. Prog. Detail Specification Part II	Comp. Prog. Module Listing	Test Plans & Requirements Specification	Handbook	Users Manual
Application Programs			X			X		
Logistics Inventory Control				X				
Resupply					X			
Usage Prediction					X			
Trend Detection					X			
Command & Control				X				
Mission Planning					X			
Mission Scheduling					X			
Mission Monitoring					X			
Orbital Ops Control					X			
Rendezvous Control					X			
Docking Control					X			
Performance Evaluation					X			
Command Signal Generation					X			
Command Conflict Resolution					X			
Message Code Formulation					X			
Message Code Verification					X			
Message Storage					X			
Experiment Data Management				X				
Real Time Process Control					X			
Data Storage & Retrieval					X			
Data Compression					X			
Data Analysis					X			
Data Display					X			
Operational Data Management				X				
Device Control					X			
Command Execution					X			
Shuttle/Module Alignment					X			
Command Verification					X			
Communication Control					X			
On Board Checkout & Control				X			X	
Monitor Program					X			
Fault Detection Program					X			
Fault Isolation Program					X			
Fault Prediction Program					X			
Recertification Program					X			
Calibration Program					X			

Table 3-2. Major Contract End Item Matrix (cont.)

Computer/Program Module \ Major CEI	Subsystem Requirements Specification	Software Operational Design Specification	Comp. Progr. Development Specification Part I	Comp. Progr. Detail Specification Part II	Comp. Progr. Module Listing	Test Plans & Requirements Specification	Handbook	Users Manual
Subsystems Operations Management	X					X		
Guidance & Control Subsystem		X		X				X
Attitude Determination					X			
Navigation Determination					X			
Maneuver Determination					X			
CMG Control					X			
RCS Control					X			
Electrical Power Subsystem		X		X				X
Solar Array Control					X			
Inverter Control					X			
Fuel Cell Control					X			
Central Power Control					X			
Current Flow Control					X			
Reaction Control Subsystem		X		X				X
Thrust Valve Control					X			
Nitrogen Quantity Balance					X			
Environmental/Life Support SS		X		X				
Gaseous Storage Control					X			
CO2 Management					X			
Atmospheric Control					X			
Thermal Control					X			
Water Management					X			
Food Management					X			
Special Life Support					X			
Structures Subsystem		X		X				
Berthing					X			
Failure Detection					X			
Crew Subsystem		X		X			X	
Crew Data Management					X			
Subsystem Status Control				X				
Subsystem Interface Control				X				

Table 3-2. Major Contract End Item Matrix (cont.)

Computer/Program Module \ Major CEI	Subsystem Requirements Specification	Software Operational Design Specification	Comp. Progr. Development Specification Part I	Comp. Progr. Detail Specification Part II	Comp. Progr. Module Listing	Test Plans & Requirements Specification	Handbook	Users Manual
Support Programs			X					
Simulation Programs				X		X	X	
Data Generation					X			
Environment Generation					X			
Operator Display Gener.					X			
Operator Control Interpr.					X			
Applic. Progr. Simulation					X			
Supervisory Pr. Simulation					X			
Utility Programs				X		X		X
Compiler					X			X
Assembler					X			X
CP Symb. Modification Pr.					X			X
Master Tape Generation					X			X
Recording					X			X
Storage Dumping					X			
Generator/Start					X			
Application Support Programs				X		X		
Personnel Records Maint.					X			
Catalog Maintenance					X			
Data Base Verif. & Maint.					X			
Information Retrieval					X			
Display Formatting					X			
Telemetry Formatting					X			
Library Routines					X			
Recording Program					X			
Reduction Program					X			

4.0 COMPUTER PROGRAM DEVELOPMENT TEST AND CONFIGURATION CONTROL PLAN

4.0 COMPUTER PROGRAM DEVELOPMENT, TEST, AND CONFIGURATION CONTROL PLAN (WBS 94012-3)

The objective of this task was to prepare a Space Station computer program development, validation, and configuration control plan which provides a significant increase in cost effectiveness over existing configuration management methods. The philosophy for computer program development specified by the plan is scientific and oriented towards engineers rather than programmers. This plan defines the scheduling, monitoring, and management requirements for program development, with descriptions of the requirements for each deliverable item and major event.

This task consisted of the following steps:

Review Configuration Management Plans - A set of configuration management plans were gathered from available sources. The plans were reviewed to determine the various approaches that have been utilized to date in the management of software production activities. A concise description, organized to facilitate a comparative analysis, was prepared. Emphasis was placed on the validation/acceptance procedures and criteria, test bed requirements, and configuration control procedures.

Review MSS Program Characteristics - The various types of computer programs required to support the Space Station were identified and analyzed to determine the contract end items to be produced. The types of programs considered were programming aids, operational and application programs, utility and executive programs, and validation/test bed programs. The analysis of the programs was presented in narrative and tabular form.

Establish Validation/Acceptance Criteria - An evaluation of the validation and acceptance criteria employed by other configuration management plans was performed. A recommendation was made for the approach most appropriate to the MSS application and was accompanied by the selection rationale. If an existing approach was suitable, it was adopted, otherwise a new one was devised.

Establish Configuration Control Plan - Configuration and change control refers to the management throughout the design, development, and operational phases of the Space Station Program. Recommendations were made for control procedures required for systematic evaluation, coordination, and approval or disapproval of all contract end items after the system specification has been prepared. These recommendations were based upon evaluation of the candidate configuration management plans for the Space Station application.

Develop Implementation Schedule - A time-phased schedule was developed to graphically depict contract end items pertaining to each of the various types of computer programs being developed for the MSS, as characterized in step two of this task. This schedule is very general as opposed to scheduling

the development of each specific Space Station program and ground support component.

Establish Test Bed Requirements - After review of the MSS computer programs, an analysis was made of the requirements for testing programs for the Space Station. A recommendation was made for the facilities required in testing the computer programs, e.g., simulation data, programming aids, simulation tools, and required equipment.

OBJECTIVE

The objective of this task was to define the conceptual structure and content of a cost-effective Modular Space Station (MSS) computer program development, a validation and configuration control plan which can be implemented to produce the computer programs for the MSS software assembly.

The plan defined by this task is to be considered as a baseline for a detailed Space Station Computer Program Development Plan which will be produced by a Computer Program Integration Control Agency upon issuance of the "go-ahead" for the MSS project.

Section 4.1 presents the assumptions used and includes a brief discussion of the status of software development planning as well as a discussion of its application to aerospace software.

Section 4.2 presents a discussion of the MSS software system to be developed. The types of software required, the organization of this software into a product structure and the identification of contract deliverable software items are defined.

Section 4.3 is an overview of the software development approach, implementation processes and facilities to be employed in the total development plan. This section defines and discusses the philosophy, division of effort, production sequence, testing activities, and management structure recommended.

Section 4.4 presents the recommended MSS software development production plan. Schedules of events, activities and end product deliveries for software production phases are presented as well as the management aspects of authority and coordination.

Section 4.5 defines the recommended software test and validation plan. The types of testing to be performed, their objectives, documentation requirements, facilities, schedules, acceptance criteria and tools to be employed are presented.

Section 4.6 presents a configuration control plan for the production and testing of the software products required. The organizational and phasing relationships of configuration control for specification maintenance, status reporting and change processing are defined.

Sections 4.4, 4.5, and 4.6 constitutes the total recommended MSS Computer Program Development Plan.

4.1 ASSUMPTIONS

The following assumptions were used in the preparation of this report.

a. All major data processing requirements and computer program components of the Data Processing Assembly (DPA) for the MSS are as defined in Part 3.0, COMPUTER PROGRAM SPECIFICATION TREE.

b. Systems Management techniques and procedures will be contractually required for the acquisition phase of the MSS program.

c. MSS software definition, design, production, validation and verification will be accomplished through more than one contractor.

d. Initial flight qualified software will be required by mid calendar year 1983 in order to conduct pre-launch system tests. Fully flight qualified software for initial operating capability will be required in the first calendar quarter of 1984.

4.2 STATUS OF SOFTWARE DEVELOPMENT PLANNING

Since 1965 all large scale software development projects have been planned with some degree of systems engineering management. The precedent for this type of planning has been the establishment of system engineering procedures for the development of weapon systems which are best exemplified by the Air Force Systems Command Series 375 manuals. In the early states of systems engineering management, software development planning activity was greatly handicapped because of the hardware orientation of the defined procedures. Considerable effort by both governmental and commercial organizations since 1966 has now resulted in the establishment of specific systems management concepts and procedures for the development and implementation of software.

Systems management planning refers to the defining and scheduling of the activities and events required to achieve a technically balanced, cost-effective total system. In general, the total plan will consist of three major parts. These are: (1) definition and production, (2) test and validation and (3) configuration control.

From a review of current software system management procedures and proposed computer program development plans, it is evident that the efforts to apply systems management to software development has resulted in a fairly uniform set of activities and events. The actual names for these activities and events vary, but the sequential schedules indicate similar time oriented functions and flows. However, technically there can be considerable variations in the structure of different plans. The principal areas where these technical differences occur are: (1) management organizations, (2) areas of authority and responsibility, (3) titles and content of required documentation, (4) degree of parallel activity, (5) end-item structures, (6) number of versions of end-products and (7) status reporting methods.

The technical differences between various computer program development plans appear to reflect two major factors. First is the degree of complexity including anticipated changes in system requirements. Second is the extent to which testing and validation must be performed prior to actual operation including the extent to which actual operational hardware will be available for testing and validating this software.

The current trend in planning the software development for very complex potentially requirement changing systems is the multi-model approach. This approach calls for the staggered but concurrent development of various versions of application computer programs, with successive versions reflecting all prior analysis, test results and requirement changes. Support computer programs, both operational and developmental, are generally developed as a single-model. For software system development requiring extensive testing and validation, the current trend is to plan for some type of centralized test center and facility for subsystem and system testing. The simulations, breadboard hardware or actual hardware on which the developed computer programs are to operate are provided at these test centers.

Aerospace computer program development, particularly the on-board segment, has been plagued with problems of schedule slippages, and high development costs for many years. Numerous studies have been conducted to define the characteristics and critical problem areas of this type of software development. Various technical and management recommendations have been made to overcome the critical problems of past aerospace software development. The most significant was the use of an application-specific, procedure-oriented higher-order programming language. Concepts of this language required that it be acceptable and usable by engineer/programmer personnel. As a result of the higher-order language recommendation, new aerospace oriented programming languages were developed and are currently available for use.

Management recommendations for aerospace programming have been concentrated in the areas of configuration control and test and validation. Configuration control processes and procedures are considered to be extremely important in order to minimize the communication breakdown that has occurred in past systems when system requirement changes were made. When appropriately applied, configuration control through standards and conventions is also considered to be the means of eliminating many past software development incompatability problems.

In most studies of the problems of aerospace software development, testing and validation was identified as the major activity where both cost and time overruns were experienced. Recommendations to overcome the critical problems of aerospace software development included the following: (1) standardization management procedures for testing, (2) establishment and control of baseline system requirements, (3) use of common software development tools and (4) more thorough definition and control of validation and performance demonstration requirements.

Current computer program development plans for aerospace applications reflect the results and recommendations made from these past problem area studies.

4.3 THE MSS SOFTWARE SYSTEM

The total software system for the space station project will include the computer programs carried on-board the modular space station, the supporting ground-based computer programs and all development and testing computer program tools. The computer program development plan presented is for the on-board portion of this total system.

Computerized data and information management functions for the Modular Space Station (MSS) will be performed within the Information Subsystem (ISS). This subsystem includes the data processor assembly (DPA), the display/control assembly, the software (computer-program) assembly and the communications assembly as shown in Figure 4-1 . The ISS will provide support to all other MSS subsystems for automation, performance monitoring, configuration control and crew operations. It will also provide the support required for experiment operations and data handling as well as external interface communications and data exchanges.

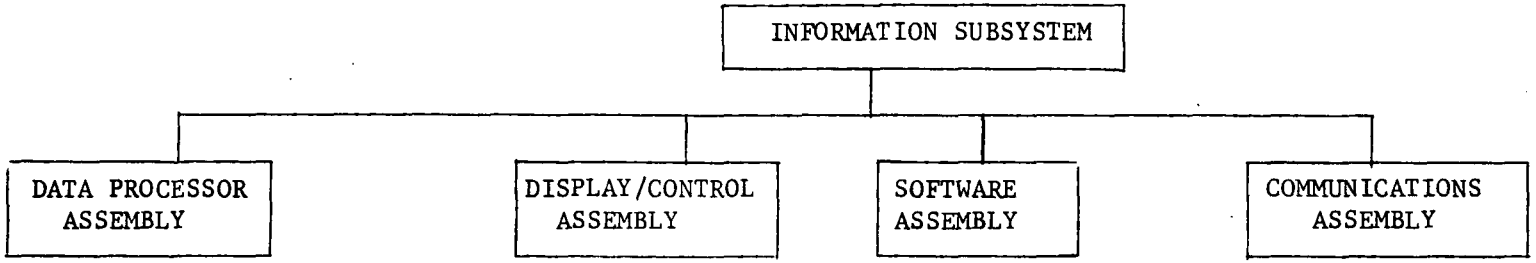
The computer programs required to perform all ISS functions, and the software tools used to develop, test and validate these computer programs constitute the MSS software assembly. Figure 4-2 identifies the five basic types of computer programs that are required and specifies various classes of operations for each type.

Types of Software. The functions of the various types of software identified in Figure 4-2 have been defined and described in Part 3.0. For development planning purposes these types of programs can be further identified by the principal software techniques which they will contain. Figure 4-3 lists these principal techniques.

A hierarchy for the computer programs to be developed for the MSS software assembly is defined in Part 3.0. Applications-type computer programs were divided into two groups. These were: (1) the application programs required for each specific MSS subsystem and (2) common application programs which are identified as application support programs. Support type computer programs were also divided into two groups and identified as (1) utility and (2) simulation programs.

In order to define a computer program development plan, the software items must be discretely identified. In structuring both hardware and software systems, extensive use of the terms SYSTEM, SUBSYSTEM, ASSEMBLY, SUBASSEMBLY, and MODULE are used. These terms have been employed in defining the basic hardware components of the modular space station and have a common meaning throughout MSS documentation. However, these same terms are commonly used to structure a software system[1]. To avoid an obvious terminology conflict with the MSS hardware structure, yet define the software structure in common used terms, the following software definitions are used:

[1] American National Standard Institute - Vocabulary for Information Processing, ANSI X3.12 - 1970.



CENTRAL PROCESSOR
- Arithmetic Processor
- I/O Processor
- Oper. Mem. Unit
- Mass Mem. Unit

PREPROCESSOR

DATA BUS

RACU

CENTRAL TIMING

ARCHIVE MEMORY

COMMAND CONSOLE
- Color CRT
- A/N Displays
- Keyboards
- Hand Controller
- M/A Displays

REMOTE TERM. UNIT
- Color CRT
- Keyboards
- M/A Displays

HARD COPY PRINTER

MICROFILM PROJ.

CMDR'S. STATEROOM CONSOLE

LOCAL M/A DISPLAYS

LOCAL CLOCK DISPLAYS

SUPERVISORY PROGRAMS

APPLICATION PROGRAMS

SUPPORT PROGRAMS

DATA BASE

TEST & VERIFICATION

SEMIDIRECT. ANT.

APPROACH RADAR

TRANSPONDER

MODULATION PROCESSOR

CENTRAL SWITCH

RECORDERS

AUDIO VIDEO UNIT

TV CAMERAS

TV MONITORS

FACSIMILE UNIT

PAGING AMPLIFIER

figure 4-1. ISS Structure

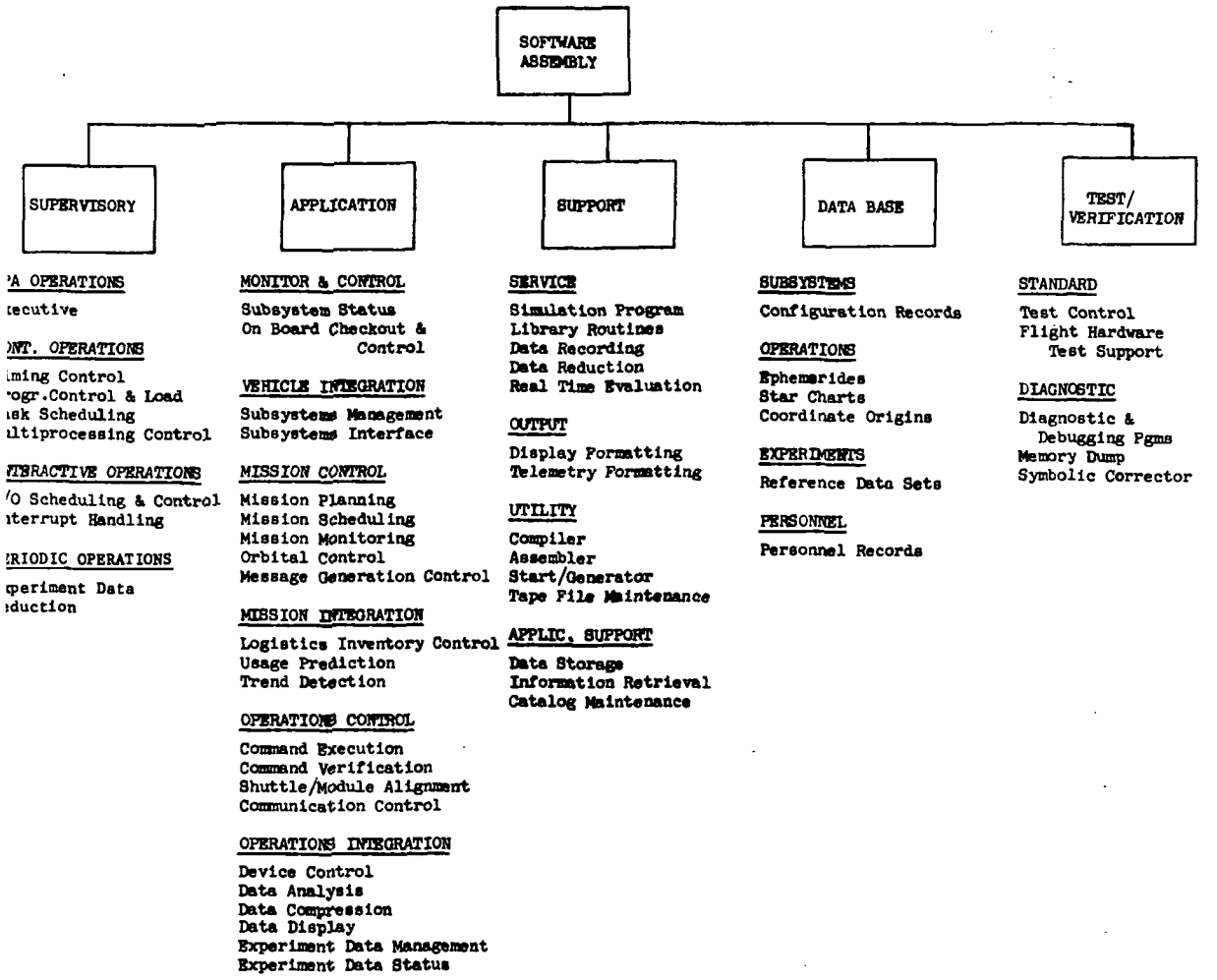


Figure 4-2. MSS Software Assembly Categorization

for each category: Define requirements for:

Language/compiler/translator identification
Word length/record length
Verification acceptance procedures
Program interaction/interfaces
Hardware interaction
Documentation
Utilization

SUPERVISORY PROGRAMS	APPLICATION PROGRAMS	SUPPORT PROGRAMS	DATA BASE PROGRAMS	TEST/VERIFICATION PROGRAMS
1. Initialization	1. Status Checkout	1. Program Generation	1. Allocation	1. Comparison
2. Decision Making	2. Status Reporting	2. Assembly	2. Address Transformation	2. Initialization
3. Time Control	3. Formatting	3. Compilation	3. Data Organization	3. Reformatting
4. Time Analysis	4. Symbol Manipulation	4. Formatting	4. Data Management	4. Time Analysis
5. Task Scheduling	5. Monitor and Control	5. Time Analysis	5. Sorting	5. Error Detection
6. Load Control	6. Bookkeeping	6. Code Improvement		6. Simulation
7. Task Allocation	7. File Manipulation	7. Simulation		
8. Configuration Control	8. Mathematical Calculations			
9. Execution Monitoring	9. Statistical Calculations			
	10. Time Sequencing			
	11. Initialization and Update			

Figure 4-3. Software Assembly - Techniques

- Computer Program Module (C.P. Module) - A single or small group of computer programs that perform a single or basic operation. A module will normally accept a single set of input parameters and produce a single set of output parameters.

- Computer Program Routine (C.P. Routine) - A group of computer program modules which perform a specific type of operations orservice a specific system segment.

- Software Subsystem - A set of computer program assemblies with common usage and/or common location in the computational hardware structure.

- Software System - The collection of software subsystems capable of performing all required ISS data processing functions.

This structure is illustrated in Figure 4-4 with a detailed breakdown, below the assembly level, shown for the supervisory C.P. assembly. A similar detailed structure exists for each C.P. assembly.

As indicated, a total MSS software system model or version will consist of four software subsystems and twelve C.P. assemblies. The number of routines and modules can only be determined following a final software systems design. Part 3.0 listed those that have been established in the conceptual design.

4.4 MSS SOFTWARE DEVELOPMENT CONSIDERATIONS

The projected ten year MSS operational life and the potential to interface with all future space oriented activities must be considered in establishing the total computer program development, test and configuration control plan. The initial objective will of course be to produce a software system capable of meeting the requirement for launch and initial operation. This software system and the plan used to produce it will constitute a baseline for all operational life changes. The plan, therefore, should provide a capability to develop and implement major and minor software system changes.

MSS initial software organizational design establishes the software module as the basic software element. The majority of all code production will be at the C.P. module level. In most cases C.P. modules will be procedure specific, and through their variable organization into C.P. routines and C.P. assemblies, flexibility of computational capability will be achieved. The software development plan must consider this module building block concept and provide for a successive sequence of increasing complexity module integration development activities.

The software development plan must also reflect the concurrent development of the MSS subsystem hardware. Concurrent development of hardware and software has resulted in problems for many past aerospace systems. Development plans did not account sufficiently for uncertainties in hardware/software trade-offs and interfaces which were changed or modified as development proceeded.

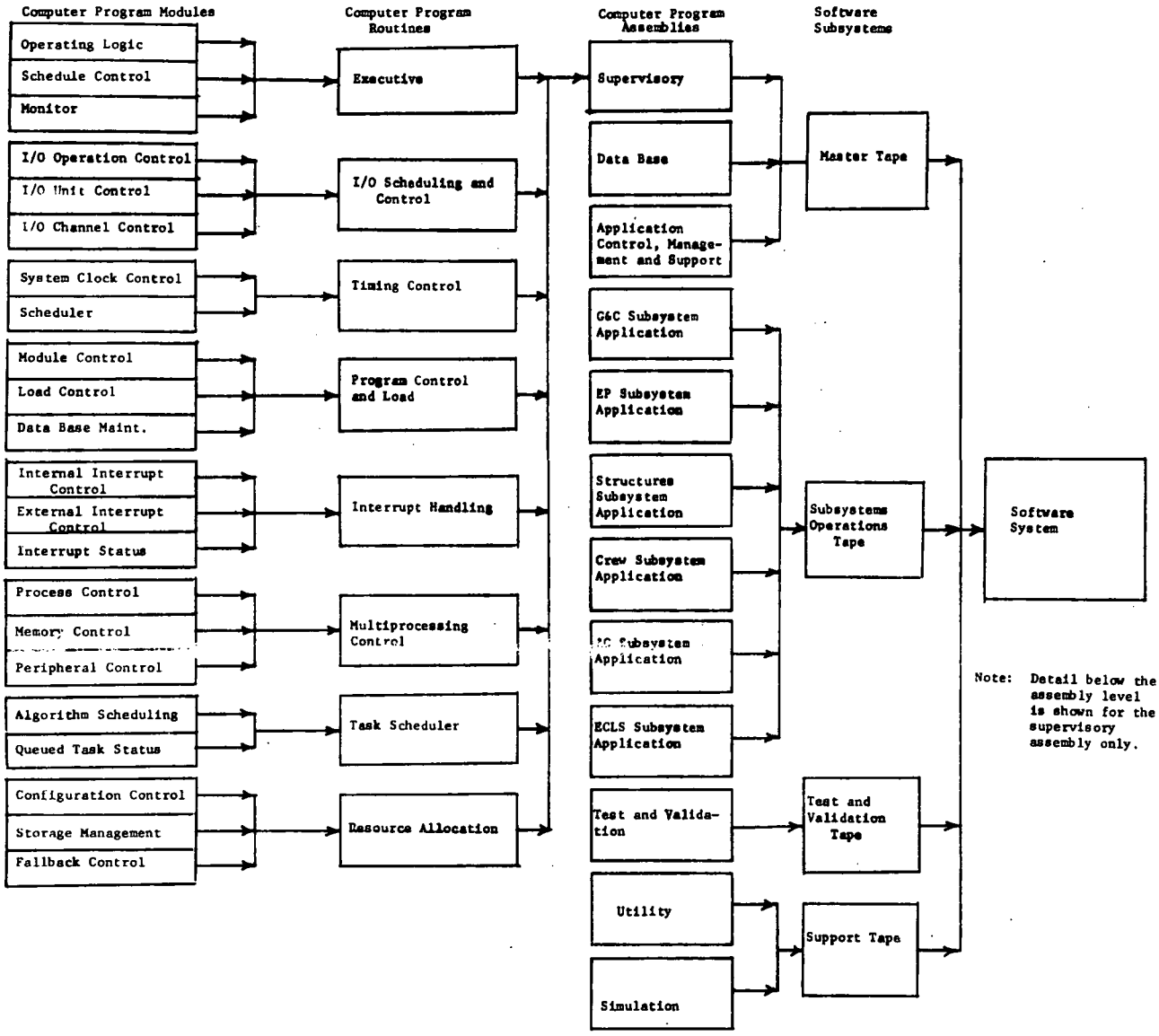


Figure 4-4. MSS Software Structure

4.5 THE MSS SOFTWARE DEVELOPMENT CONCEPT

This section presents an overview of the total development plan for the MSS software to be produced and tested. Major software elements are defined and their production relationships with other software elements presented.

Cost-effectiveness is the primary objective of the MSS Computer Program Development, Test and Configuration Control Plan. Based upon the experience of previous aerospace software development, it is evident that to be cost-effective, the plan must provide for the following:

1. Early demonstration of the software structure.

2. Early validation of system logic and software specifications.

3. Early feedback of test results even if preliminary and/or incomplete.

4. Capability to accept new requirements, improved techniques and procedures with minimum impact on schedules.

5. Permit flexible, responsive, configuration control.

6. A division of effort which maximizes the commonality and functional similarity of the software to be produced.

7. An efficient test process through all states of software buildup.

The basic approach philosophies recommended to best achieve the above described desired conditions are:

1. Assign full software system development, integration and testing responsibilities to a single software system management contractor. This contractor should be responsible for all software related MSS project coordination and should have the authority to subcontract software production of modules and routines and specific testing functions.

2. Division of production, integration and testing effort should be based on a best mix of functional and operational commonality. Functional commonality relates to similar mathematical and/or data processing techniques. Operational commonality relates to system modes such as real-time, off-line, occasional usage, critical or non-critical operations.

3. A multi-model approach for MSS subsystem application software should be followed for those MSS subsystems where extensive hardware development is required. The multi-model approach consists of staggered but overlapping software production of various versions of a specific software product.

4. Five levels of progressive testing should be employed with subdivisions of these levels for subsystem and final system testing. These testing levels should be:

   . C.P. Module - stand alone testing

   . C.P. Routine - stand alone testing

   . C.P. Assembly - integrated testing

   . Software Subsystem - integrated testing

      a. Simulations

      b. Hardware interfaced

   . Total Software System

      a. Interrupt real-time

      b. Ground based real-time

      c. On-board real-time

5. Established software configuration control procedures should be employed and modified where required.

Figure 4-5 presents the proposed MSS software system authority and coordination structure. The major difference between this and previous structures is the establishment of major contractors for hardware and software under a system's management contractor. Each of these contractors would have total responsibility for the management of the development and integration of their respective areas. Integration of the hardware and software into a final operational system would remain the responsibility of an overall system's management contractor.

The MSS software System Management and Integration Contractor would have full responsibility for the following principal items:

1. A final computer program development test and configuration control plan.

2. Procuring or in-house production of all computer program modules and routines.

3. Approving software contractor module and routine test plans and procedures and witnessing these tests to verify acceptability. For in-house produced modules and routines an independent testing contractor or non production oriented in-house organization would be used.

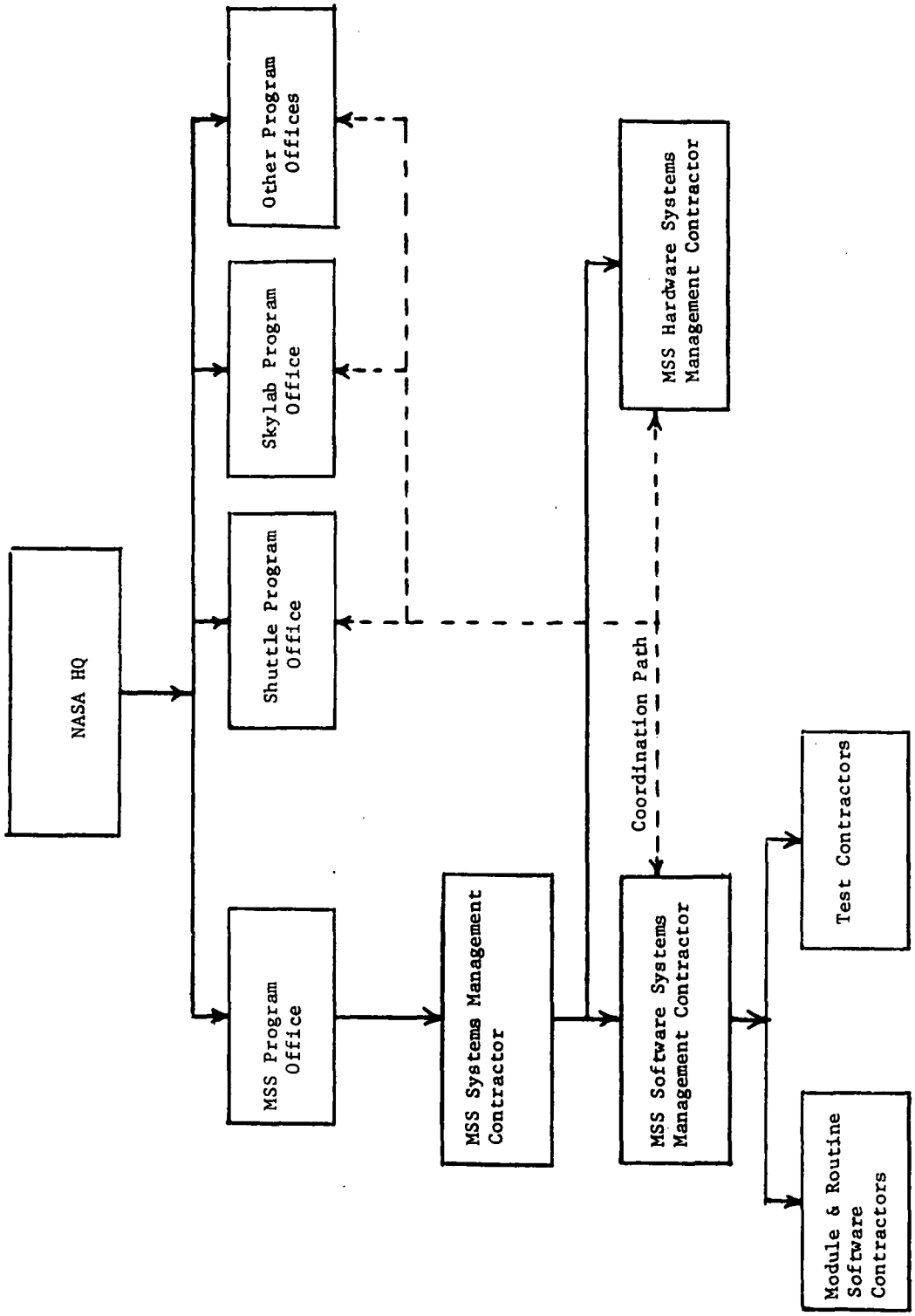


Figure 4-5. MSS Software System Authority and Coordination

4. Act as the software configuration control coordinator by providing all required inputs to the MSS Configuration Control Board and maintaining and issuing all required software configuration control documentation.

5. Provide all required MSS software system coordination between other space projects, MSS hardware development and the overall MSS project.

6. Develop and operate a Computer Program Development Facility (CPDF) for the purposes of: (1) assembly testing, (2) subsystem integration and (3) interrupt real-time system testing.

7. Approve all software test plans and procedures for assembly and system tests conducted at the MSS Project Operated Centralized Test Facility (CTF). Witness all software oriented tests at the CTF.

8. Plan and conduct through the MSS operational structure the on-board real-time final system qualification tests. Through its responsibilities and activities, the MSS Software System Management and Integration Contractor will implement the approach philosophy itemized.

Except where specific software module development can be enhanced by a particular software contractor's capabilities or experience the routine level, (groups of related modules) should be the minimum level for software subcontracting. For procurement purposes, all functional routines as defined in Part 3.0, and partially illustrated in Figure 4-4, should be further subdivided into classes or operational modes. Figure 4-2 lists the principal classes or operational modes for each of the five types of MSS computer programs.

The reason for operational mode subdivision of the routines is to structure MSS computer programs into groups, regardless of function, having different development requirements. For example, supervisory computer programs required for continuous operations should be developed and tested to a higher degree than those required for periodic operations. The process is further emphasized if these two groups are further divided into groups for critical and non-critical operations.

For software production and testing efforts, the extent to which a module is to be operationally used and the extent to which it is required for critical and non-critical operations will determine the degree to which its production will be monitored and how it will be tested. The following four categories of software moduels would be identified in the production process.

1. Continuous operation - critical
2. Continuous operation - non-critical
3. Periodic operation - critical
4. Periodic operation - non-critical

To minimize the impact of new and changing requirements on program schedules, a multi-model approach for application software will be followed. Three versions of a given MSS software assembly should be adequate to accommodate all anticipated changes and ensure that overall program goals are met.

Version one (V_1) allows for the establishment of a baseline from program design requirements and specifications developed during the conceptual phase.

Version two (V_2) will result from updates and refinements of the V_1 requirements obtained from analytical and functional simulations. Version V_2 will be verified by subsystem tests at the Computer Program Development Facility and the Centralized Test Facility.

Version three (V_3) software will be created to include all revised requirements obtained from implementing version V_2 and from changes due to hardware development and/or mission requirements. Version V_3 will be oriented to the full requirements for on-board MSS flight testing.

The relationship of the three versions of a software package are illustrated in Figure 4-6. The multi-model approach will be applied to the MSS MASTER TAPE SUBSYSTEM and the MSS SUBSYSTEMS OPERATIONS TAPE. A single-model development of the TEST AND VALIDATION SUBSYSTEM TAPE and the SUPPORT SUBSYSTEM TAPE is considered to be adequate (See Figure 4-4).

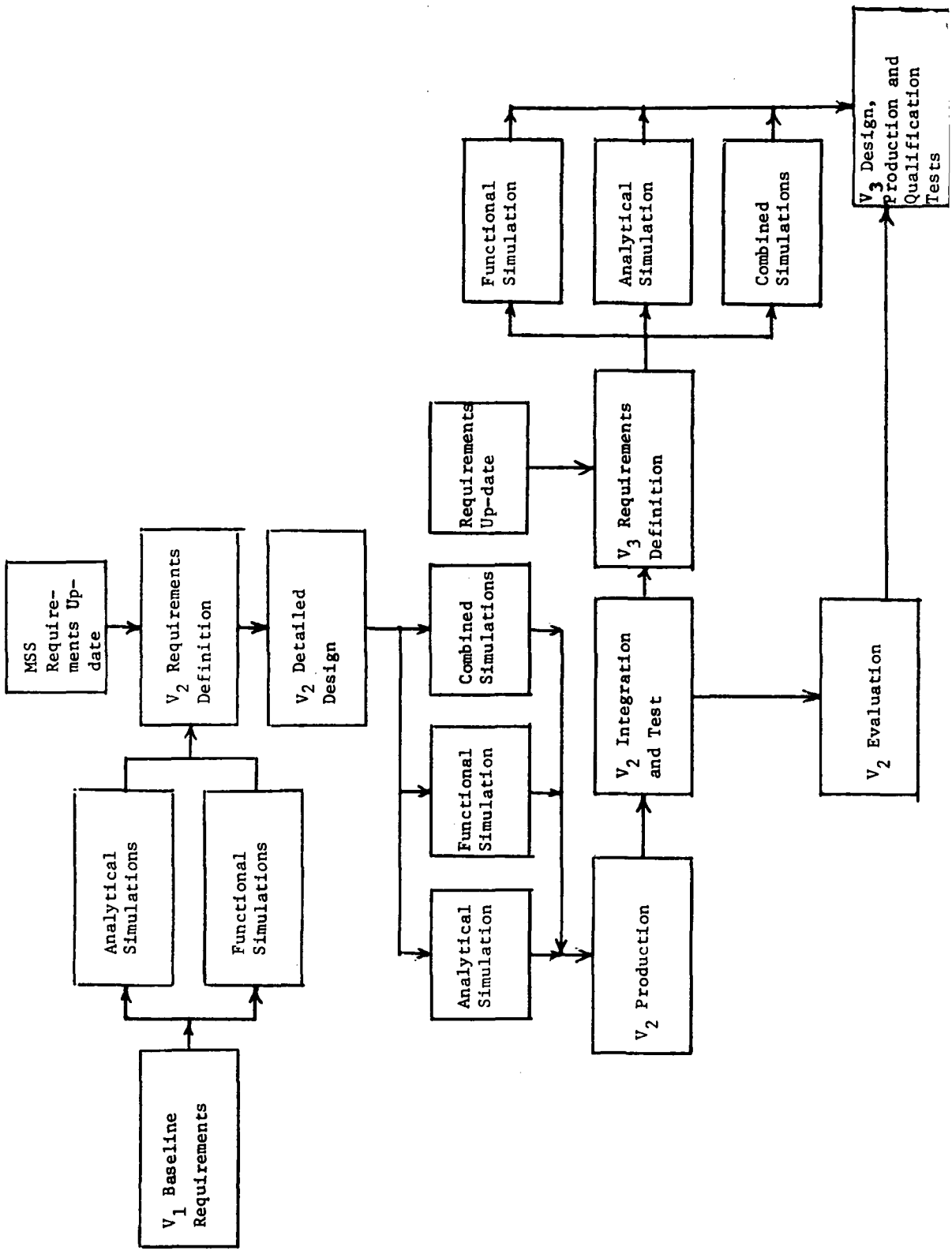
The approach taken to integrate and test the implemented MSS software involves an orderly buildup of tests and an organized methodology to assure the meeting of all requirements. Test and evaluation will be accomplished through five levels of tests which are:

1. C.P. Module Tests

   These are stand alone tests performed on each individual module. Early verification of basic logic to meet specifications will be established.

2. C.P. Routine Tests

   These are stand alone tests of combinations of related modules. They will provide early demonstration of the software structure.


V_1 Baseline Requirements
Analytical Simulations
Functional Simulations
MSS Requirements Update
V_2 Requirements Definition
V_2 Detailed Design
Analytical Simulation
Functional Simulation
Combined Simulations
V_2 Production
Requirements Up-date
V_3 Requirements Definition
V_2 Integration and Test
V_2 Evaluation
Functional Simulation
Analytical Simulation
Combined Simulations
V_3 Design, Production and Qualification Tests

3. C.P. Assembly Tests

These are tests of integrated related routines. These tests verify both the functional operation and the organizational structure of groups of modules. Interface operation is exercised through simulated inputs.

4. Software Subsystem Integration Tests

The integration process and operational capability of the MSS subsystem program tapes are conducted with both simulated and actual inputs (where practical). These tests are the first step in qualifying operational ready software.

5. Total Software System Tests

These are validation and evaluation tests of the total MSS software system. Three stages of total systems testing will be performed. First in interrupt real-time with simulated input conditions; second in real-time on actual and simulated hardware and, finally on board the MSS in its operational envrionment.

The approach to MSS software configuration control is to adopt the basic practices and procedures of current software configuration control management. This methodology has been reviewed for various governmental and commercial activities and has been formulated into a configuration control plan.

4.6 DEVELOPMENT PROCESS

Functional analysis encompasses those activities used in achieving definitive system, subsystem and functional unit performance requirements. This includes all major analysis and design studies used to formulate specifications. Substantial functional analysis has been performed in the conceptual design phase of the MSS. This effort will continue in varying degrees throughout the software development process.

Fundamental analysis through simulation will be used to verify and test the software system design through its eight basic development phases described in the next subsection.

The MSS software implementation process will consist of eight basic phases. These phases overlap in time and feedback from later phases are provided to earlier ones in order to implement changes in concept and/or design. The eight basic software implementation phases are:

1. Software System Design
2. Computer Program Requirements analysis
3. Computer Program Design
4. C.P. Module coding and testing
5. C.P. Routine checkout and testing
6. C.P. assembly and subsystem integration and testing
7. Software System Tests and Evaluation
8. Software System Validation and Acceptance

The complexity of the MSS software system requires the utilization of interactive and multi-model approaches.

MSS software system development will be highly dependent on simulations. Except for the flight test, MSS hardware and software will be operated with one or more portions of the system being simulated. A large number of simulations will be used and developed for the MSS program. Some will be simple computational aids to allow rapid solution of complex problems, others will be sophisticated closed-loop simulations requiring large amounts of computer capacity.

A majority of MSS simulations will be used for functional analysis in the design process. These will, in general, be programmed on general purpose computers provided by the Software System Management and Integration contractor or individual software contractors. Special simulations involving operational hardware, special environments and complex interfaces will be provided at a Centralized Test Facility.

The full range of simulations required for MSS software development requires additional effort to specify their exact number and requirements.

4.7 DEVELOPMENT FACILITIES

With the exception of special integrating computer programs and simulations, all MSS computer programs will be produced by software contractors at their respective facilities. This includes both operational and support software. Specialized simulations required for integration on the Computer Program Development Facility and the Centralized Test Facility will be produced at these facilities.

The Computer Program Development Facility (CPDF) for MSS will be developed and operated by the Software Systems Management and Integration contractor. It will provide laboratory and office facilities required to integrate and test MSS assemblies, subsystems and the total system.

Major test facilities will consist of:

1. Software contractors facilities for module and routine testing.

2. The CPDF and the Centralized Test Facility (CTF) for assembly subsystem and system tests.

3. The CTF and the MSS for system acceptance tests with MSS on-board testing constituting final acceptance testing.

4.8 MSS SOFTWARE DEVELOPMENT PLAN

Deliverable software - As discussed elsewhere, the total MSS software system will consist of four subsystem tapes, twelve major assemblies and substantial number of routines made up of modules. The modules, routines, assemblies and sub-systems will constitute Computer Program Contract End Items (CPCEI) for a given total system. CPCEI's will be equivalent to Computer Program Configuration Items (CPCI) as defined in the configuration control plan.

For production scheduling the MSS software structure presented in Figure 4-4 must be structured into items for the multi-model approach. Additional subdivision on the basis of continuous or occasional usage will also be made to identify continuous operation software, which if delayed in production, could result in propagated delays for integrated testing.

The purpose of multi-model software development is to provide a formalized process to incorporate changes in requirements and techniques with a minimum of disruption to the software production schedule. Therefore, all MSS software items which are related to flight hardware, where potential change is highest, will be developed through the multi-model approach. Table 4-1 lists the MSS software assemblies that will develop through the multi- and single model approaches. This structure will result in multi-model development of the MASTER SUBSYSTEM and the SUBSYSTEMS OPERATION TAPES. All modules and routines for those assemblies developed under the multi-model approach will follow the procedures of the approach.

Three of the multi-model developed software assemblies listed in Table 4-1 are affected substantially by the operational mode of the system. These are: the supervisory assembly, the data base assembly and the application control assembly. Production of these assemblies will be divided into subassemblies as follows:

a. Supervisory Assembly

   1. DPA Master Executive Subassembly
   2. Continuous Data Bus Subassembly
   3. Interactive Operations Subassembly
   4. Periodic Operations Subassembly

b. Data Base Assembly

   1. Subsystem Data Subassembly
   2. Operations Data Subassembly
   3. Experiments Data Subassembly

c. Application Control, Management and Support Assembly

   1. Mission Control Subassembly
   2. Vehicle Integration Subassembly.

Table 4-1
MSS Multi and Single Model Developed Software Assemblies

Multi-Model Developed Software Assemblies	Single-Model Developed Software Assemblies
1. Supervisory	1. Test and Validation
2. Data Base	2. Utility
3. Application Control, Management and Support	3. Simulation
4. G&C Subsystem	
5. E.P. Subsystem	
6. Structures Subsystem	
7. Crew Subsystem	
8. R.C. Subsystem	
9. ECLS Subsystem	

The MSS software development process will be divided into a series of eight phases as shown in Figure 4-7. The phases do not represent firm divisions of time but overlap to provide feedback so that changes can be implemented. This process is further supplemented by software configuration control as described in Section 4.6.

The eight phases consist of the following:

1. Data Processing System Design - results in system and subsystem requirement specifications.

2. Computer Program Requirements Analysis - results in preliminary specifications at the routine and module levels.

3. Computer Program Design - results in detailed routine and module specifications.

4. C.P. Module Code and Test - results in coded and stand alone tested modules.

5. C.P. Routine Assembly and Test - results in assembled groups of modules functionally tested.

6. C.P. Assembly and Subsystem Integration and Test - results in blocks of integrated and interactive software functionally and operationally tested.

7. Software System Test and Evaluation - results in a complete, integrated data processing system tested as far as practical in a ground based complex.

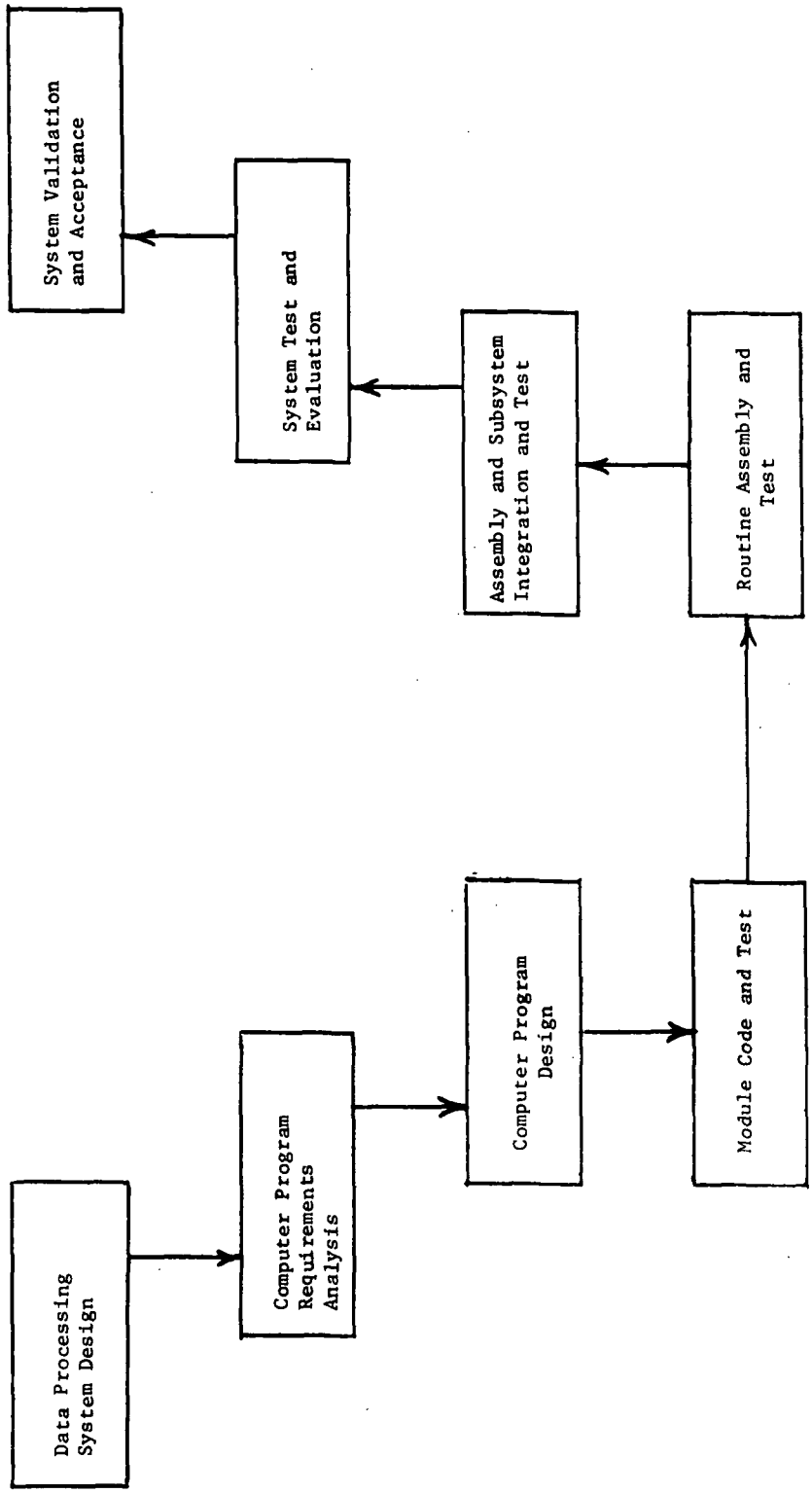


Figure 4-7. Software Development Phases

8. Software System Validation and Acceptance - results in an accepted complete system operating on-board the space vehicle.

The multi-model process requires that requirement specifications be developed and published for three distinct versions of all software items developed via this means. Version V_1 will be based on the studies conducted during the conceptual design of the MSS and updated to include design reviews up to the start of the MSS Development Phase. V_1 specifications will be issued two months following initiation of MSS development.

Software versions V_2 and V_3 will be developed with a fair degree of overlay in order to assure that the overall MSS software schedules are met.

The specifications for version V_2 will begin at the end of month four and continue through month ten. When released, these specifications will be configuration controlled and formal production initiated. The V_2 model will be carried through subsystem testing. Preliminary design reviews of version V_2 will be initiated in the sixth month, one month after release of the first V_2 series of specifications.

Version V_3 software specifications will be released at the beginning of the sixteenth month and will be the controlled version of the final operational software. Version V_3 will follow the same production steps as version V_2 but will be system tested.

The basic multi-model development process is illustrated in Figure 4-8.

The single model development process will be used primarily to develop MSS support software. Software items developed under this approach will be from a single version of specifications which are modified or changed through the configuration control process.

Initial operating versions of single model developed software will be required by the eighteenth month in order to perform multi-model version V_2 testing. Revision and changes to these initial versions, if required, will be made by the twenty-seventh month in order to perform multi-model V_3 testing.

The MSS software system is based upon the concepts of modularity. Each module will be a thoroughly defined unit which meets specific objectives. Modules will be organized into routines having responsibility for technical performance. A uniform module development process will be followed by all module producing contractors.

The development cycle for each module, regardless of version, will follow the steps shown in Figure 4-9.

Module development is the lowest level of development. Modules must be produced to correspond to the software functional specifications. Module testing will be performed by the software development contractor with performance verified by the software management contractor.

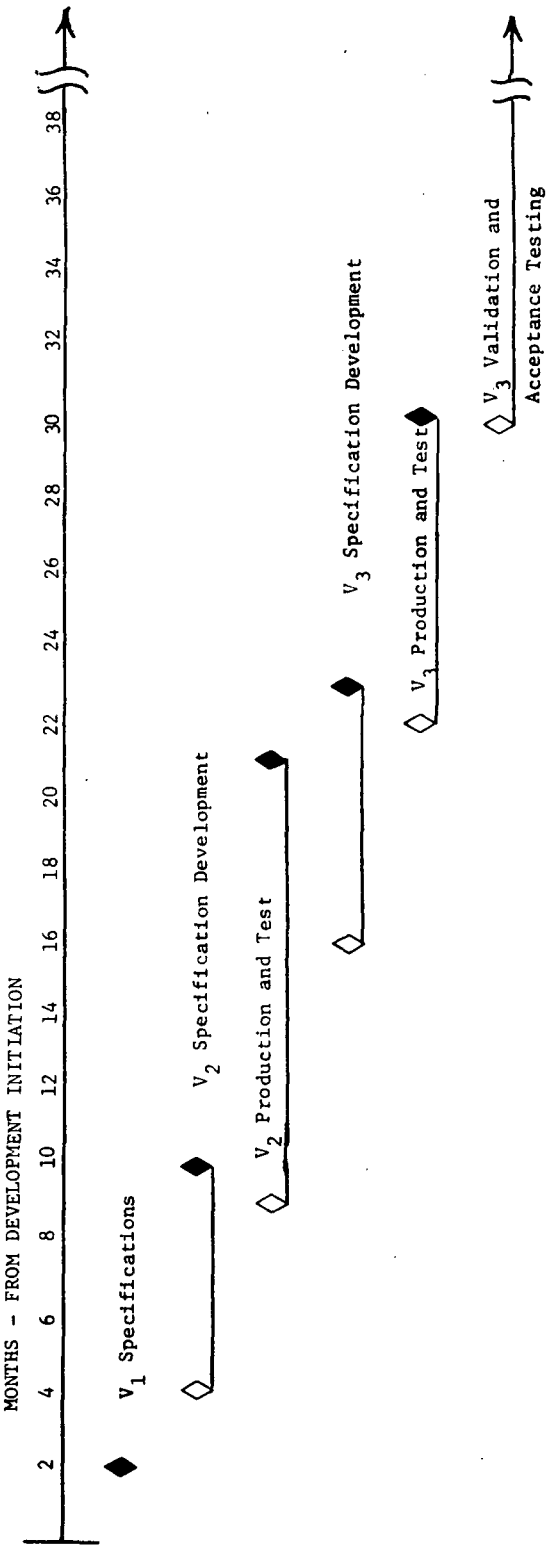


Figure 4-8. Multi-Model Process

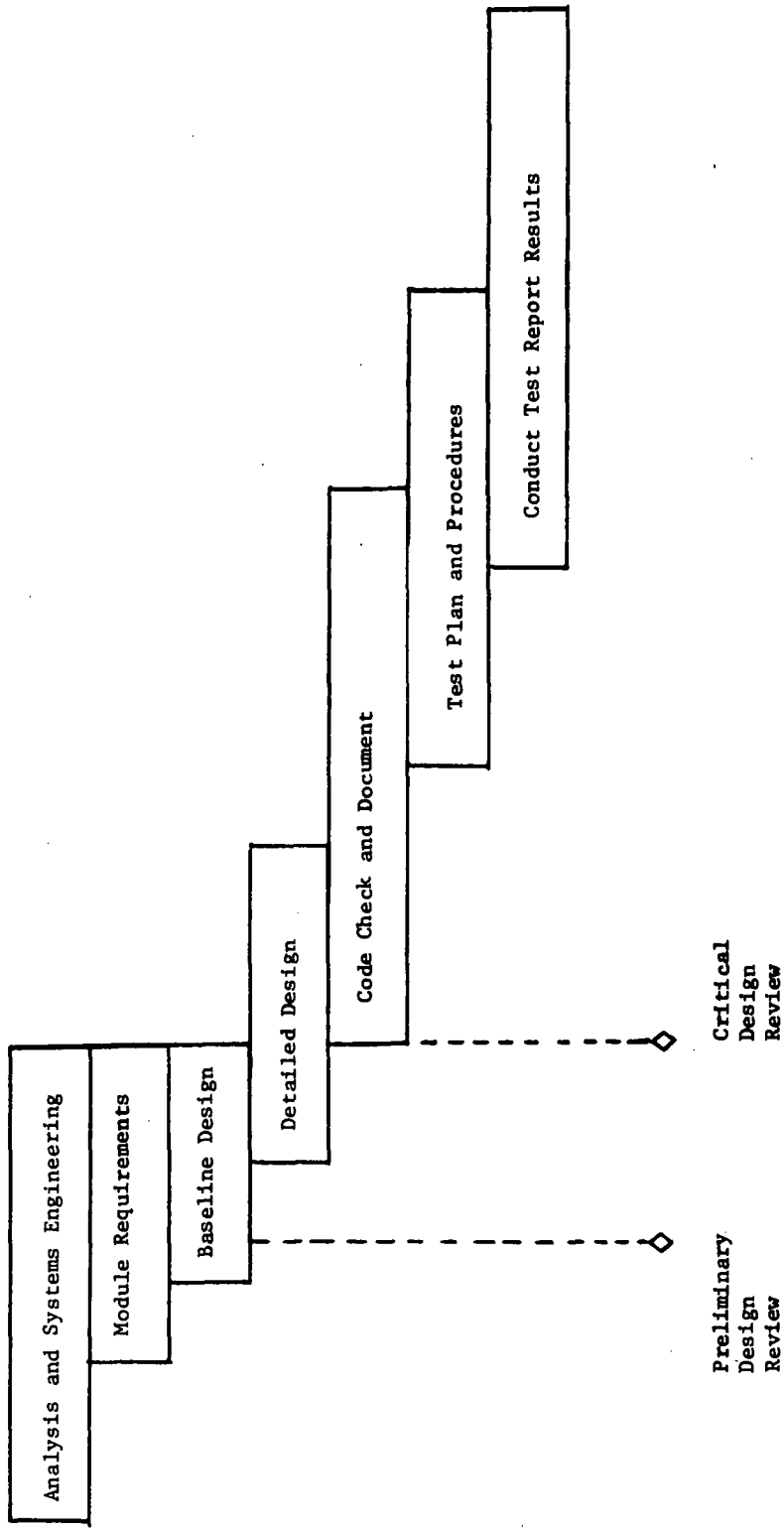


Figure 4-9. Module Development Process

The preparation of detailed, high quality documentation is essential to the software development process. Part 3.0, this volume, has defined and described the software documentation hierarchy to be used for MSS software development.

4.8.1 DEVELOPMENT SCHEDULE

The current proposed development schedule for the Modular Space Station is presented in Figure 4-10. Preliminary system analysis and design studies will be performed to mid 1977. Data processing requirement studies will commence in mid 1975 and will result in finalized baseline specifications at the end of 1977. Software development and production to produce a total system will cover the two and one-half year period from 1978 to mid 1980. Total MSS system integration and acceptance tests will be performed through 1981. System installation, checkout and launch preparation will require one year with completion scheduled by early 1983. Launch and in-flight qualification tests will be performed through 1983. Mission operations with a fully qualified and accepted MSS will commence in early 1984.

MSS software system development up to system testing must be completed in the allotted two and one-half year period. Interrupt real-time and real-time ground based total system testing will be performed during the MSS System integration and acceptance period. Final qualification testing will be performed following launch.

Gross software development events and activities required to meet the overall development schedule are presented in Figure 4-11. Single model developed software tools for on-board software development and testing are scheduled to be initially available for multi-model developed version V_2 assembly and subsystem testing. Multi-model developed version V_3 and all single model developed on-board software will be available for integration and acceptance testing by mid 1980.

Major integration and acceptance testing will be performed and completed by the start of 1982. However, this effort will continue until launch to support installation and checkout.

4.8.2 MANAGEMENT

The basic organizational structure for software development was presented in Part 4.5. Total responsibility for the development and implementation of the MSS software system will rest with the Software System Management and Integration Contractor. This contractor will have the authority to contract for the procurement of module and routine tested software items. It will be responsible for assuring that these items meet all requirements and for integrating and testing these items as assemblies and subsystems. This contractor will also be responsible for all system tests but may subcontract protions of this activity.

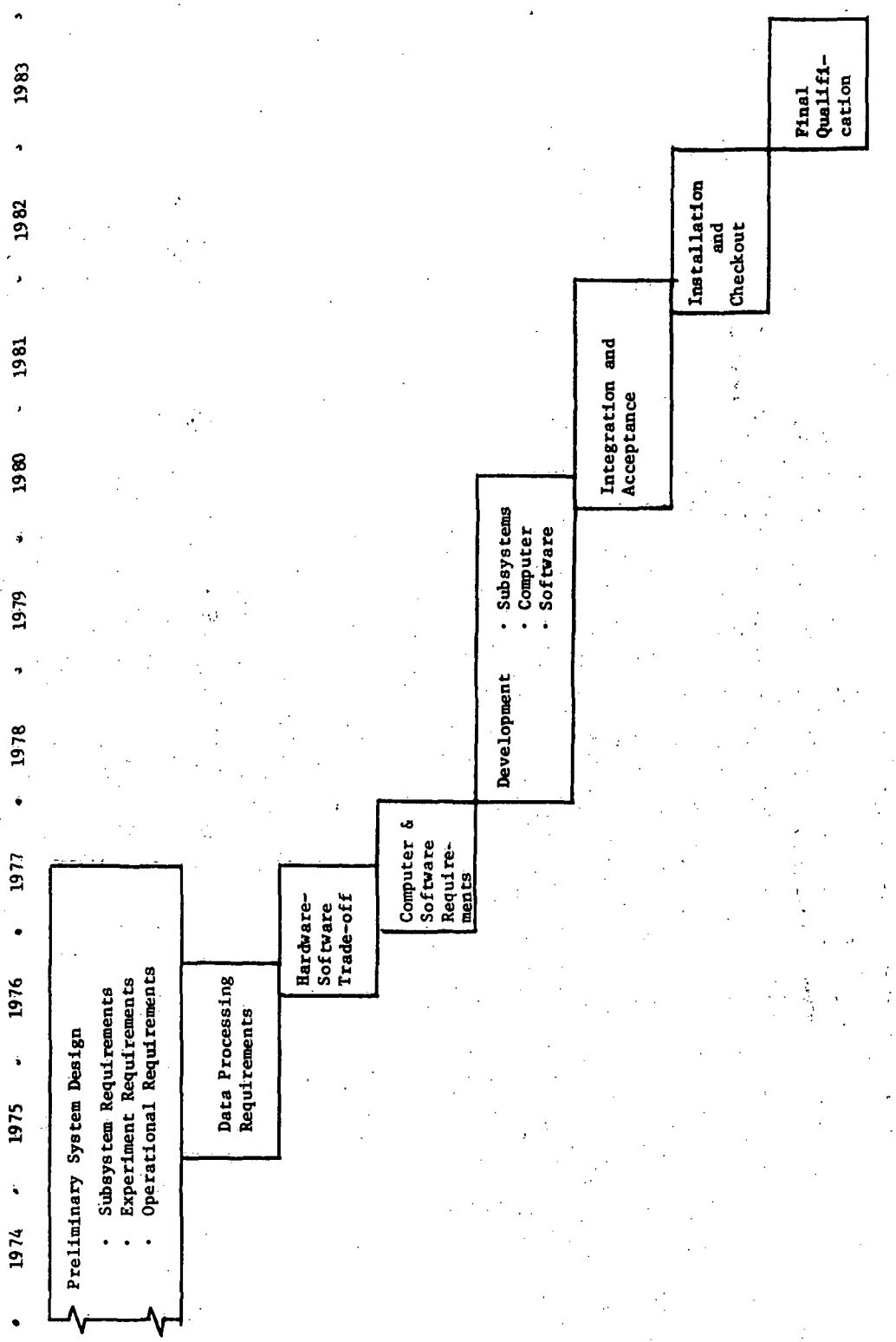


Figure 4-10. MSS Development Schedule

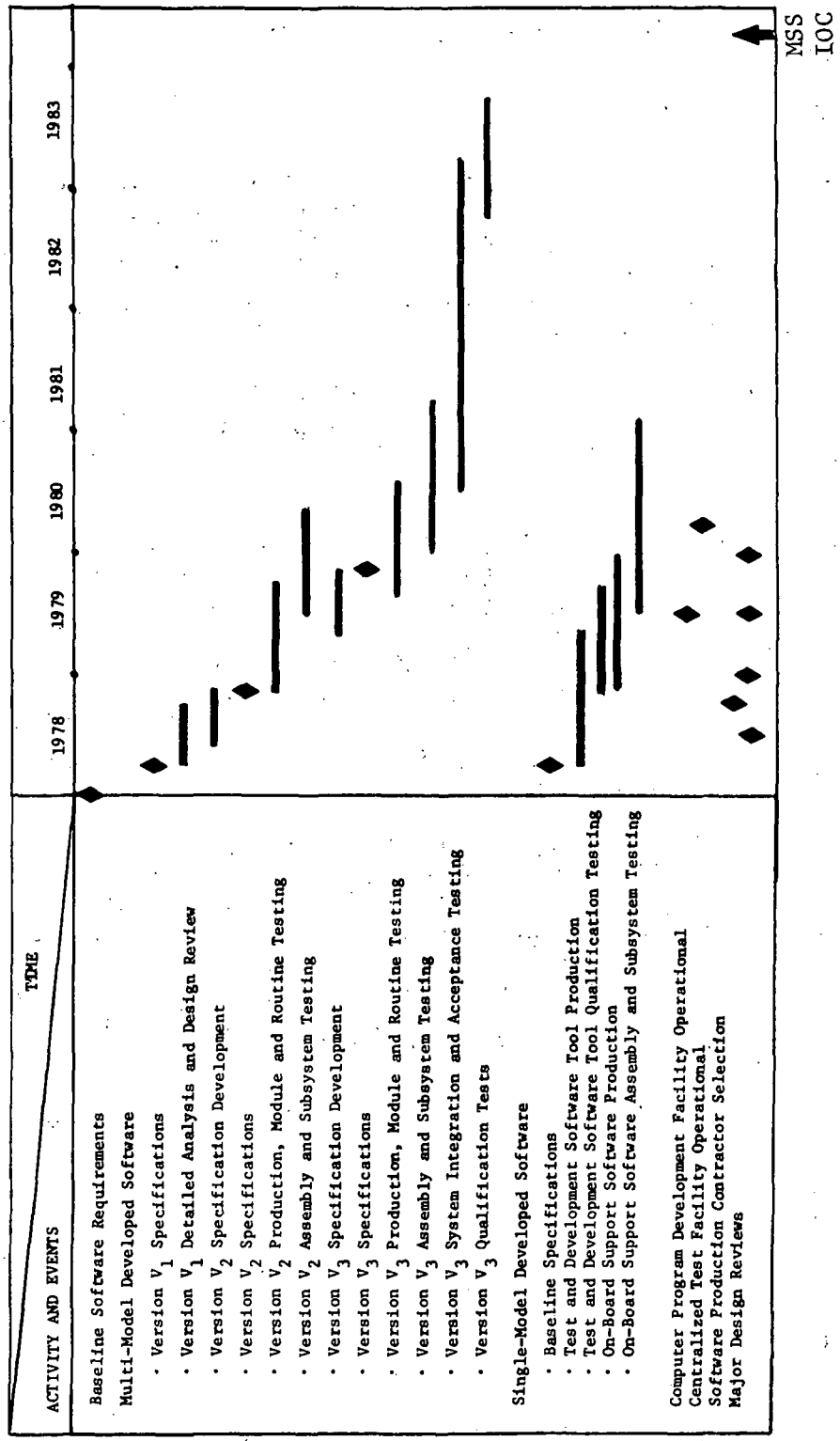


Figure 4-11. MSS Software Development Schedule

All MSS software related activities will be coordinated by the Software Management Contractor. This includes both internal and external activities.

4.9 MSS TEST AND VALIDATION PLAN

The primary objective of the computer program test and validation plan is to ensure that the MSS software assembly performs in accordance with its requirements. The computer program test and validation plan presented in this section is an integrated approach which provides for continuing validation of the specifications and detailed design through simulation as well as detailed testing of the implemented hardware and software.

The validation approach comprises three component activities that interleave and provide continual feedback and iteration of specifications of the design. These activities are illustrated in Figure 4-12 and are described as follows:

1. The software design approach is validated through functional simulations which provide information on subsystem execution time, resource utilization, allocation and logic.

2. Specifications are validated through combined analytical and functional simulations which model the operating system software. This approach permits early validation of the subsystem and development specifications and can be used for correlation with conceptual analytical simulations.

3. Design implementation is validated via an integrated test plan which utilizes a building block concept. Tests begin with the smallest item of configuration controlled software (the module) and then combine these by functional tasks and groups, until the entire system is integrated and run in a realistic operational condition. System evaluation tests that stress full design load capabilities will be performed in both real-time and interrupted real-time in order to demonstrate total system performance.

The Test and Integration Plan (TIP) will be accomplished through five primary levels of testing. These are:

1. Level 1 - C.P. Module Tests

   These are stand alone tests performed on each individual module. They will test module performance up to the full design requirements.

2. Level 2 - C.P. Routine Tests

   Routine tests are tests of combinations of related modules. These tests will be performed in isolation from other routines. Like module testing, full functional design requirements of a routine plus internal module interfaces will be tested.

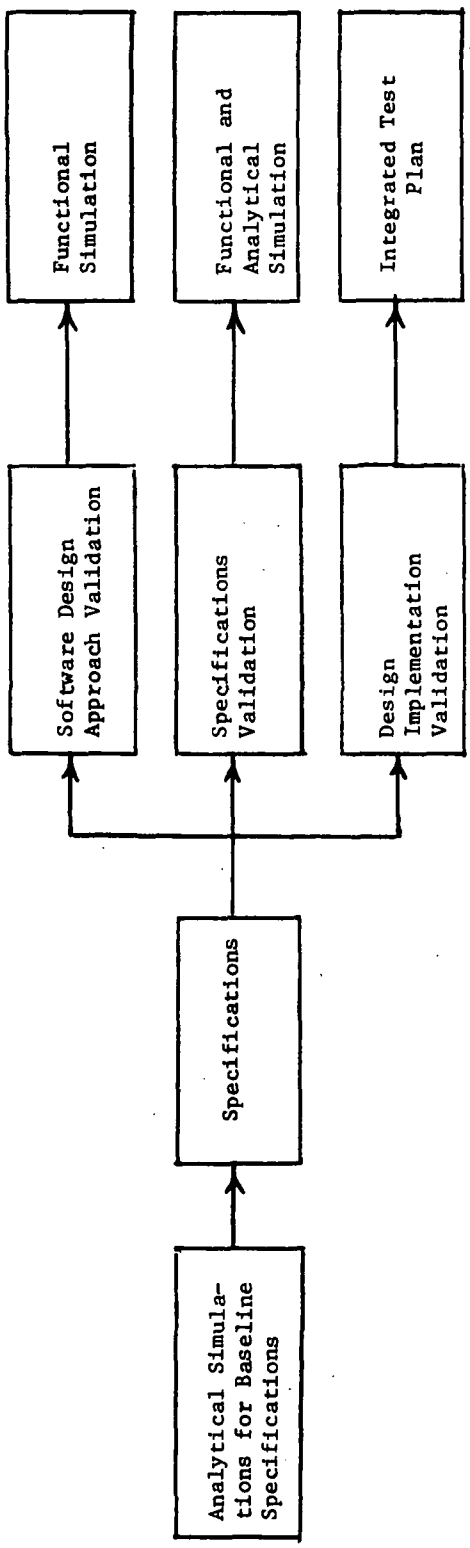


Figure 4-12. MSS Software Validation Phases

3. Level 3 - C.P. Assembly Tests

Assembly tests are tests of integrated related routines. For each assembly both the functional and interface requirements of the modules and routines will be tested. Major interfaces with other assemblies will be via actual hardware, breadboarded hardware and/or digital simulations.

4. Level 4 - Software Subsystem Integration Testing

These tests combine groups of assemblies (previously tested) into major protions of the MSS software system. Their primary purpose is to validate the software readiness of an operational system.

5. Level 5 - Software System Integration and Evaluation Testing

This is a test of the complete MSS operational system. It will exercise all aspects of the system under normal and heavy loads. Its purpose is to validate the MSS software system for operational deployment. Initial tests will be performed in interruped real-time and subsequent testing will be performed in real-time for acceptance.

Figure 4-13 illustrates the basic levels of the Test and integration plan. Details on the objectives, required documentation, tools, facilities and criteria for each of these testing levels are presented in the following subsection.

4.9.1 TEST STRUCTURES

4.9.1.1 C.P. Module Testing

Objectives. The objective of module testing is to demonstrate and verify that the functional capability of the module meets design requirements. The tests will be stand alone tests on individual modules conducted by the contractor responsible for the module production.

Documentation. Module test documentation shall consist of a test plan, a description of test procedures and a report of results. These items shall be structured within the Computer Program Configuration Item (CPCI) documentation requirements for configuration control.

Tools. All development and testing software tools used by a contractor to produce and test a software module shall be identified in test documentation. Special software tools required for testing shall be considered as support software for the module and shall constitute a part of the module.

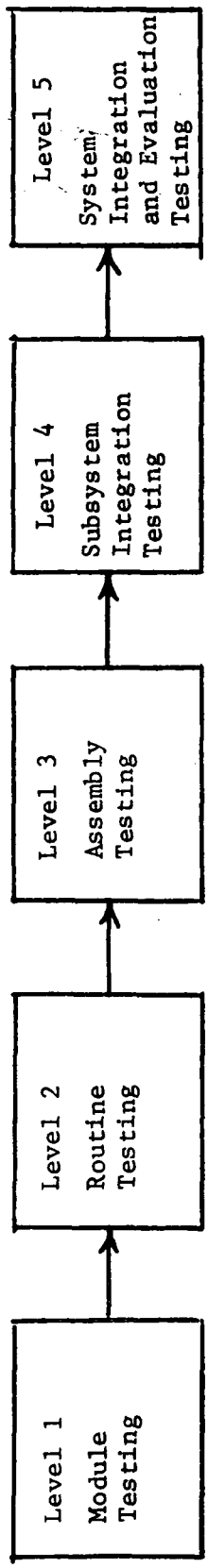


Figure 4-13. Levels of Testing and Integration

Facilities. MSS module computer programs will be developed and tested on facilities provided by the software contractor responsible for their production. The MSS Software Integrating Contractor shall verify the adequacy of the facilities and shall witness all tests performed.

Validation and Acceptance Criteria. Module functional capability shall be determined through parameter testing which ensures that all major decision points, defined in the requirements, are executed and that all outputs successfully compare with expected outputs.

4.9.1.2. C.P. Routine Testing.

Objectives. The objectives of routine testing are to test the functional and internal interface capabilities of one or more organizationally related modules. The property of these related modules of a given routine being structured into a group of computer programs capable of performing functions requiring multiple modules is demonstrated by these tests.

Documentation. A test plan, test procedures, description and test results report shall be published for each tested routine. The test results documentation for each module within the routine shall reference or include the individual module test results.

Tools. Any special software testing tools required for routine testing shall be developed by the contractor responsible for the tests or provided as part of the modules which make up the routine.

Facilities. Those software contractors responsible for the production of a specific routine and its modules shall provide the facilities to test the routine. Routines in which the modules are produced by different software contractors shall be tested on facilities provided by the software integrating contractor. Individual software module contractors shall witness routine tests in which their respective modules are exercised. The software integrating contractor shall verify the adequacy of other routine testing facilities and shall witness testing at these facilities.

Validation and Acceptance Criteria. MSS software routines shall be validated and accepted on the basis of their ability to perform correctly as isolated units. Iterative parameter and data testing to include all routine module combinations will be performed.

4.9.1.3. C.P. Assembly Testing.

Objectives. Assembly testing objectives are to test the functional operation of an assembly and to test the interaction between both routines and modules within the assembly. Executive controls, I/O controls, availability of data, and timing will be extensively tested.

Documentation. A test plan and procedures document plus a test results report shall be prepared for each assembly.

Tools. Interpretive and functional simulations of hardware subsystems and other software assemblies will be used where required to meet test objectives as well as actual and breadboarded hardware.

Facilities. Assembly testing will be performed at a MSS Computer Program Development Facility (CPDF) and/or at a MSS Centralized Test Facility (CTF). The CPDF will be provided and operated by the MSS software integrating contractor. The facility shall include basic MSS system and subsystem interpretive and functional simulations. The CTF will be provided and operated by the Systems Management Contractor. The facility will provide breadboarded and actual hardware as well as simulations needed to conduct hardware/software tests. All software assembly testing shall be performed by the software integrating contractor.

Validation and Acceptance Criteria. Assembly testing represents the first level of software system integration. All functional and operational requirements of the modules and routines which constitute an assembly must be tested for both internal and external interfaces. Validation and acceptance criteria for each assembly shall include successful demonstration of the following:

- Module, routine interactions
- Data availability and organization
- Executive control interface
- Correct functional operations
- Proper timing

4.9.1.4. Software Subsystem Integration Testing

Objectives. A MSS software subsystem is a set of computer program assemblies with common usage and/or common location in the computational hardware. The objective of subsystem testing is to verify the integration of all computer program modules, routines and assemblies which constitute a software subsystem. The organizational structure and formats as well as the content of all software assemblies will be verified and compiled into a subsystem magnetic tape.

Documentation. Test documentation shall consist of a subsystem integration test plan, and procedures. The test result report will consist of a complete subsystem description including magnetic tape file and record descriptions.

Tools. Software compilation, generation and assembly tools will be used to integrate, verify and produce the subsystem magnetic tapes.

Facilities. Subsystem integration and testing will be performed by the MSS software integrating contractor on the CDPF facility.

Validation and Acceptance Criteria. Software subsystem validation and acceptance shall be based upon a totally operationally exercised subsystem with an itemized comparison of the structure and content of the subsystem magnetic tape against each C.P. assembly listing.

4.9.1.5. Integrated Software System Testing

Objectives. The primary objective of integrated system testing is to validate the total MSS software system for operational readiness. All aspects of the software system must be exercised for nominal and out-of-limit conditions. Three stages of testing will be employed in MSS integrated system testing. These will be: (1) Interrupted Real-Time testing, (2) Real-time ground testing and (3) On-board real-time testing. The objective of the Interrupted Real-Time tests will be to demonstrate functional and operational performance with simulated external interfaces. The objective of the real-time ground testing is to demonstrate operational capability on actual and breadboarded hardware. Formal acceptance is the objective of the real-time on-board tests.

Documentation. All prior module, routine, assembly and subsystem test documentation will be used to define the test plans for system testing. Test procedures and results as required for configuration control and project management will be produced.

Tools. See Facilities

Facilities. The Computer Program Development Facility and all software tools used in its operation will be used to conduct the Interrupt Real-time phase of Integrated system testing.

Real-time ground integrated system testing will be performed at the Centralized Test Facility. To the fullest extent possible, actual hardware will be used for all tests. When not available, breadboarded or simulated hardware will be used.

On-board real-time integrated system testing will be performed on the modular space station after launch and during the various stages of buildup.

Validation and Acceptance Criteria. Formal final acceptance shall follow successful and satisfactory operation of the on-board real-time tests as specified in a software flight test plan.

Preliminary validation and acceptance will follow the successful completion of the real-time ground system tests and the demonstration that all system aspects have been exercised and perform as specified.

4.9.2 TEST, VERIFICATION AND ACCEPTANCE SCHEDULE

Based upon the development schedule of Figure 4-11, major test, verification and acceptance events will be as shown in Figure 4-14.

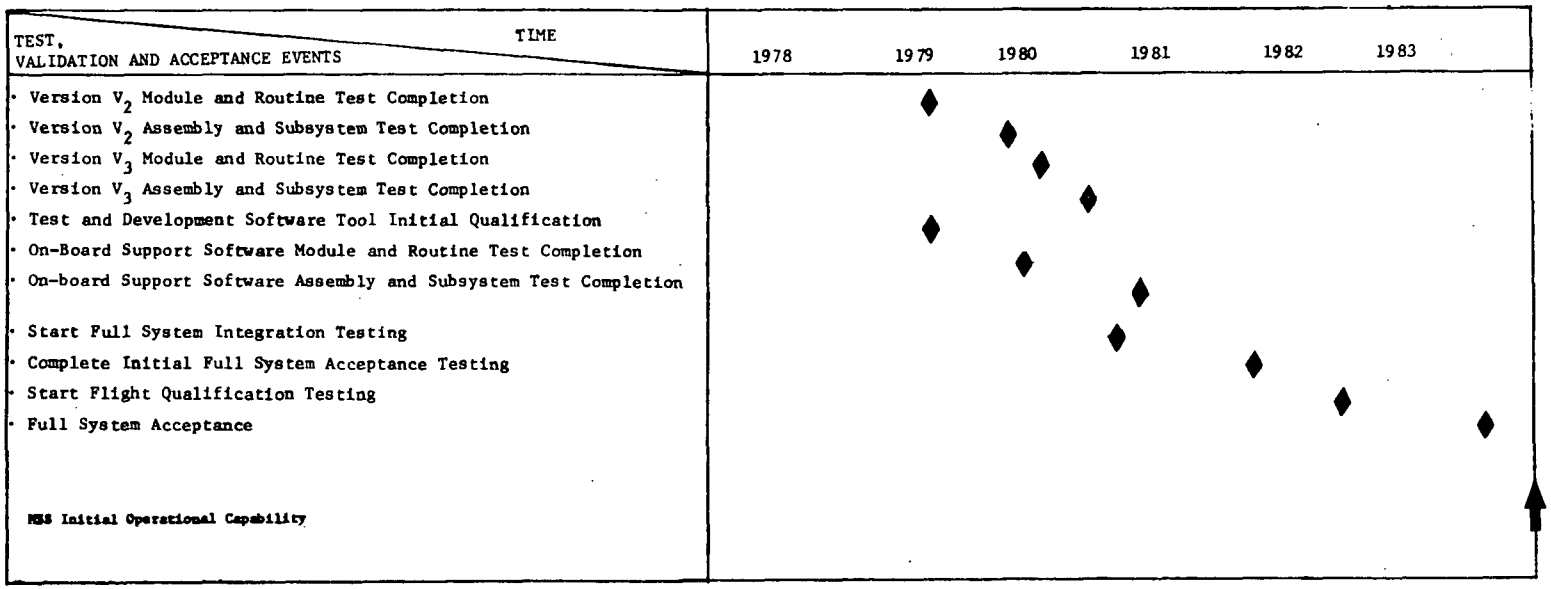


Figure 4-14. MSS Software Test, Validation and Acceptance Schedule

4.9.3 TEST FACILITIES

Three major software testing facilities will be used to develop and qualify the MSS software system.

The Computer Program Development Facility (CPDF) is primarily a software integration facility for assemblies and subsystems. However, it will also have the capability to perform total system testing in interrupt real-time with simulated inputs. This facility will be developed and operated by the Software Systems Management Contractor. The facility can also be used for module and routine software modification when required to meet integration problems. Environmental and hardware simulations will be developed for this facility.

The Centralized Test Facility will be located at the MSS hardware assembly and integration complex. This test facility will provide actual and breadboarded hardware for software system and subsystem testing. Initial real-time ground based qualification testing of the total software system will be performed with this facility. The Software Management Contractor shall have the option of developing and operating this facility or of procuring an independent contractor to be responsible for the task. Development and implementation of this facility will require close coordination with the overall MSS Systems Management Contractor to assure the availability of hardware and realistic hardware/software interfaces.

The Modular Space Station itself prior to being qualified for operation will serve as the final software test facility. Full operational capability will be demonstrated before final acceptance.

4.10 CONFIGURATION CONTROL PLAN

This section outlines a proposed approach to software configuration control for the MSS. It does not attempt to define all detailed procedures. It has been assumed that following a review of the general concepts, principles and terms presented, that detailed implementation will be a subsequent effort for the final computer program development plan.

4.10.1 FACTORS AND CONCEPTS

A computer program configuration item (CPCI) is an individually identifiable computer program or group of computer programs, for which a definite level of software specification, control, support documentation, acceptance criteria and status reporting can be defined. MSS computer program modules, routines, assemblies, subsystems and systems will be designated as CPCI's if they meet the above criteria.

All CPCI's shall be subject to full management control of all changes to established baselines. Control at the allocated baseline level will be maintained for the life of each item. Control of changes will be based on provisions equivalent or similar to those defined in DOD MIL-STD-480, CONFIGURATION CONTROL-ENGINEERING CHANGES, DEVIATIONS AND WAIVERS.

Specifications defining approved configurations will be maintained through controlled procedures to reflect all approved changes. Procedures implemented for this purpose will also include a system of accounting for the maintenance of other documents associated with computer program development and use, when changes to such documents are occasioned by approved changes to the computer program specifications. Periodic status reports will record the status of document issuance and maintenance, the status of proposed changes, and the status of delivered computer program versions.

The following terms are of special importance in configuration control of software. Where necessary, additional terms shall be defined and published.

a. Computer Program

A sequence of coded instructions and data, designed to cause a digital computer to perform a desired function, or set of functions. In general use, the term does not necessarily imply a level of complexity, size, nor equivalence to a configuration item.

b. Computer Program Configuration Item (CPCI)

An aggregation of computer program (software) elements which satisfies an end-use function and is designated for configuration management. The CPCI is an end-use item, as distinct from its specification, normally in recorded form on punched or magnetic cards, tapes, etc.; however, the configuration of the CPCI as defined in its specification does not include characteristics of the physical medium for recording and delivery. Progressive versions of a CPCI may evolve in the development process. Discrete identification by version is required. (See item e.).

c. Software

As used herein, the term "software" is synonymous with "computer programs". It does not include documentary items associated with computer program development or use.

d. Subprogram

A major part of a CPCI, identified for purposes of convenience in specifying and developing a complex CPCI as a set of subordinate elements. Subprograms are structural parts of an item.

e. Version

The actual configuration of a CPCI which is introduced for installation and test or operation into the system, in the form of a magnetic tape, card deck, etc. An existing version is modified whenever any change is made in the computer program instructions or data content.

e. continued

A new version is created: (a) when a newly developed item is first delivered for installation; or (b) when the item undergoes a significant modification, e.g., resulting from Class I changes; or (c) when indicated by an accumulation of small changes and so identified by the responsible test or operational activity.

4.10.2 APPLICABLE SPECIFICATIONS

Configuration control of MSS computer programs will involve the types and levels of specifications defined in Part 3.0. That report defines the computer program specification hierarachy and details the content of the components of the hierarchy. The applicable specifications as specified are as follows:

a. MSS Computer Development Plan
b. Subsystem Requirements Specifications
c. Software Operational Design Specifications
d. Computer Program Interface Specifications
e. Computer Program Development Specification
f. Configuration Management Specification
g. Validation Test Plans and Requirements
h. Computer Program Detailed (CPCI Part II) Specifications
i. Handbooks
j. User's Manuals
k. Test Procedures
l. Computer Program Listings
m. Program Maintenance Manuals
n. Installation Procedures

4.10.3 ORGANIZATIONAL RELATIONSHIPS

Software configuration management will involve the following major organizational elements.

a. Program Office - NASA
b. Systems Management Contractor
c. Software Integration Contractor
d. Software Development Contractors

Implementation of initial design and changes to MSS CPCI's will be accomplished by analysis and software design units of the systems management and software integration contractors. The software integration contractor will be responsible for internal controls, preparation and coordination of formal design change proposals, implementation of approved changes, specification maintenance and configuration status reporting of all MSS CPCI's.

The software integration contractor will have total responsibility for the management of all MSS CPCI integration and testing. Activities will encompass integration of items furnished by individual software contractors into the system, and the conducting or monitoring of all testing. Activities will also include: the analysis of problems and proposed changes, management of CPCI interfaces, disseminating software standards and the maintenance of configuration information.

A software configuration control board (SCCB) will serve as the action agency for the systems management contractor. It will control software-only changes but will support the configuration control responsibilities of the systems management contractor in all matters affecting hardware/software interfaces. The organizational structure and the operation of the SCCB will be a responsibility of the systems management contractor with the software integration contractor responsible for implementing decisions of the board.

4.10.4 PHASING RELATIONSHIPS

Effective software configuration control must be phased with computer program development and testing activities and with the documentation required for these activities. Figure 4-15 illustrates Configuration Control Phasing Interrelations with Computer Program Development, Test and Documentation.

a. CPCI Definition

Computer program configuration item definition is performed during the establishment of system criteria and the analysis of systems requirements. System and subsystem specifications provide the basis for this activity.

b. CPCI Preliminary Design

This activity determines the functional requirements for each CPCI in respect to: system limits, interfaces, data rates, data base requirements and checkout procedures.

c. CPCI Detailed Design

This design activity establishes, in detail, the CPCI functions to be programmed, the data organization required, internal and external interfaces and limitations or constraints. A detailed flow chart of the item is developed.

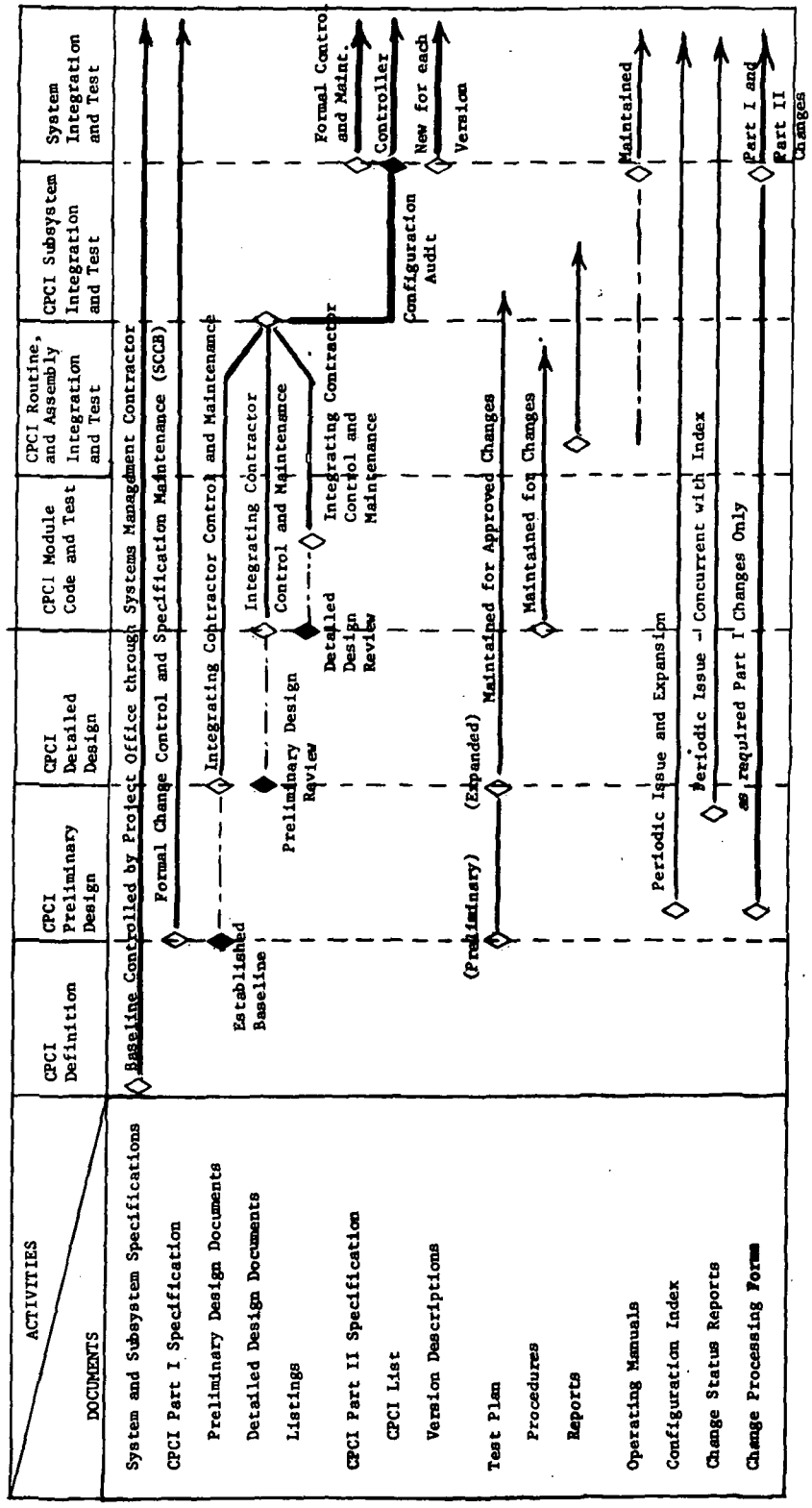


Figure 4-15. Configuration Control Phasing Interrelationships

d. CPCI Module Code and Test

This activity is the actual production of object code modules. It also includes the testing of these modules for functional performance.

e. CPCI Routine and Assembly Integration and Test

This activity is a testing and analysis effort in which the first stages of software integration are performed and the integrated units tested for functional and interface compliance.

f. CPCI Subsystem Integration and Test

Assembly integration into subsystem master tape CPCI's are performed in this activity phase. Testing of both subsystem segments and total subsystems are performed. Initial acceptance of CPCI's by the project office through performance demonstrations are an end point of this activity.

g. System Integration and Test

Activity involves total system testing and performance demonstration. Formal acceptance of a software system version is the end point of this activity.

The primary functions of the documents identified in Figure 4-15 are as follows:

a. System Specifications

- Define performance and design requirements for the MSS system as a whole.
- Identify major MSS subsystems
- Allocate requirements to subsystems
- Define Subsystem interfaces
- Provide the highest level criteria against which system testing is to be performed.

b. Subsystem Specifications

- Expand the definition of performance and design of subsystems
- Identify major subsystem components
- Allocate requirements to major components and define inter-component interfaces.
- Define criteria for testing an integrated subsystem

c. CPCI Part I Specifications

- Provide a detailed definition of inputs, outputs, and performance requirements for the CPCI as a whole.
- Identify individual CPCI functions and subfunctions
- Define input, output, and processing requirements for each function/subfunction.
- Define data base, interfaces, timing, sequencing, and requirements for special features.

d. CPCI Part II Specifications

The Part II specification for each developmental CPCI will consist of elements corresponding to those described in the Part I specification. However, emphasis will be placed on accuracy and sufficiency of content as a current definition of the CPCI at the time of delivery and acceptance, plus quality assurance requirements for testing and acceptance. Following successful completion of an audit by the SCCB, controls administered by the SCCB will assure continued maintenance of the Part II Specification to reflect all approved changes.

e. Testing Documentation

Testing of computer programs will occur within the framework of established requirements and plans for testing at successively higher levels of integrated software.

Test plans will normally be prepared in preliminary form concurrently with preparation of the Part I specification. They will be subsequently expanded and updated to reflect preliminary design of the CPCI, to permit firm scheduling of tests in relation to development and assembly schedules for individual subprograms.

The software integrating contractor will be required to assure the preparation and submission for advance approval of a Test Procedures Document for each formal test session. Test Reports will be required for individual sessions and to summarize the completion of tests for each computer program item.

4.10.5 CHANGE PROCESSING

To be effective, configuration control must be integrated with management of the development/test processes previously outlined, and must be able to handle the variations which can occur among individual CPCI's. Considering the nature of the development phase, and the magnitude of MSS software elements, changes can be expected to occur in large volume. Controls must be able to

handle this volume efficiently, without impeding technical efforts to exploit the inherent flexibility of computer programs in meeting system requirements. At the same time, it is essential that functional and design characteristics be controlled and known at all times, for the entire complex of MSS software. Elements of a recommended control process, and examples of its application under varied circumstances are summarized briefly below.

The Class I designation will apply to changes affecting NASA controlled baselines. These will be proposed by the Systems Management Contractor via Engineering Change Proposals (ECPs) and Specification Change Notices (SCNs) in accordance with established requirements. Changes to computer program configuration item baselines will be further subdivided for purposes of internal organizational control into (a) Class IIA -- changes requiring approval by the SCCB prior to implementation, and (b) Class IIB - - minor changes which a design activity may implement without prior approval, subject to review for classification.

a. Design Change Request (DCR)

The DCR will be issued as a standard form, together with preparation instructions, as the vehicle by which any participating agency may bring a suggested change to the attention of the SCCB. Action taken by the SCCB may include acceptance, rejection, or deferment. Acceptance is followed by assignment to the responsible engineering activity, and may constitute only provisional approval of the change. For Class IIA changes, formal approval will be based on the Change Proposal (CP) subsequently submitted by the responsible agency.

b. Change Proposal (CP)

A standard form will be used by all responsible agencies for proposing changes to CPCI's. The change proposal shall identify the total impact of the proposed change in respect to functional capability, schedule modifications and the resources required to accomplish the change. Sufficient detail must be provided in order for the SCCB to evaluate the proposed change and reach a decision on its disposition.

c. Change Report (CR)

Class IIB changes will be accomplished by responsible agencies without prior approval. However, all such changes will be reported to the SCCB, reviewed for classification, and fully accounted for in procedures of document maintenance and status reporting.

The need for a change to a CPCI baseline may arise from many sources. During development, changes to the Part I specification may be indicated by further analysis, changes in interfacing or higher level specifications, or other. Due to the advanced development nature of some MSS items, certain incremental changes may be programmed in advance. During system integraton and test, change requirements will typically result from analysis of installation, assembly, and test results. Often, computer program changes can be made to resolve difficulties encountered with other system elements.

Whether resulting from internal determination or by SCCB directive, each Class IIA change to an established CPCI baseline will be initiated formally through preparation of a CP. Content of the CP will be based on necessary technical study, determination of total impact, and coordination with other MSS activities affected. Following submission to the SCCB, the CP may be approved for implementation, disapproved, deferred, or returned to the design activity for further study and revision.

Any CP which impacts a higher level specification will be accompanied at the time of its initial submission by a Specification Change Notice (SCN) covering or containing exact wording of the proposed specification revisions. In the case of CPCI's, however, the distribution of SCNs with exact wording changes normally constitutes the end point of a controlled change process for computer programs, rather than the beginning. This is true because the successful completion of specification changes implies that (a) at the Part I level, all necessary CPCI analysis and definition effort has been expended, and (b) at the Part II level (as built), the computer program change has been designed, coded, incorporated into the CPCI, and tested. Hence, a CP addressing the CPCI specification at either level, or both together, will define the proposed change in sufficient detail to provide a basis for SCCB decision and to guide the necessary definition and/or computer program development efforts, but need not be accompanied at the time of initial submission by SCNs.

It is to be expected that changes at the Part I specification level will occur frequently during the design phase, prior to completion of development for individual CPCIs. Once the product baseline is established, most Class IIA changes will be combinations, for the reasons that (a) any change to the Part I will require a change to the CPCI and its controlled Part II, and (b) Class IIA changes affecting computer program design and coding should be relatively rare. Clearly, the impact of any given approved change to the Part I specification during earlier stages of design will increase progressively as development proceeds towards completion.

A synopic illustration of the Class IIA change process is provided in Figure 4-16. The diagram is highly simplified with respect to timing and inclusion of events, and should be interpreted in the light of the following comments.

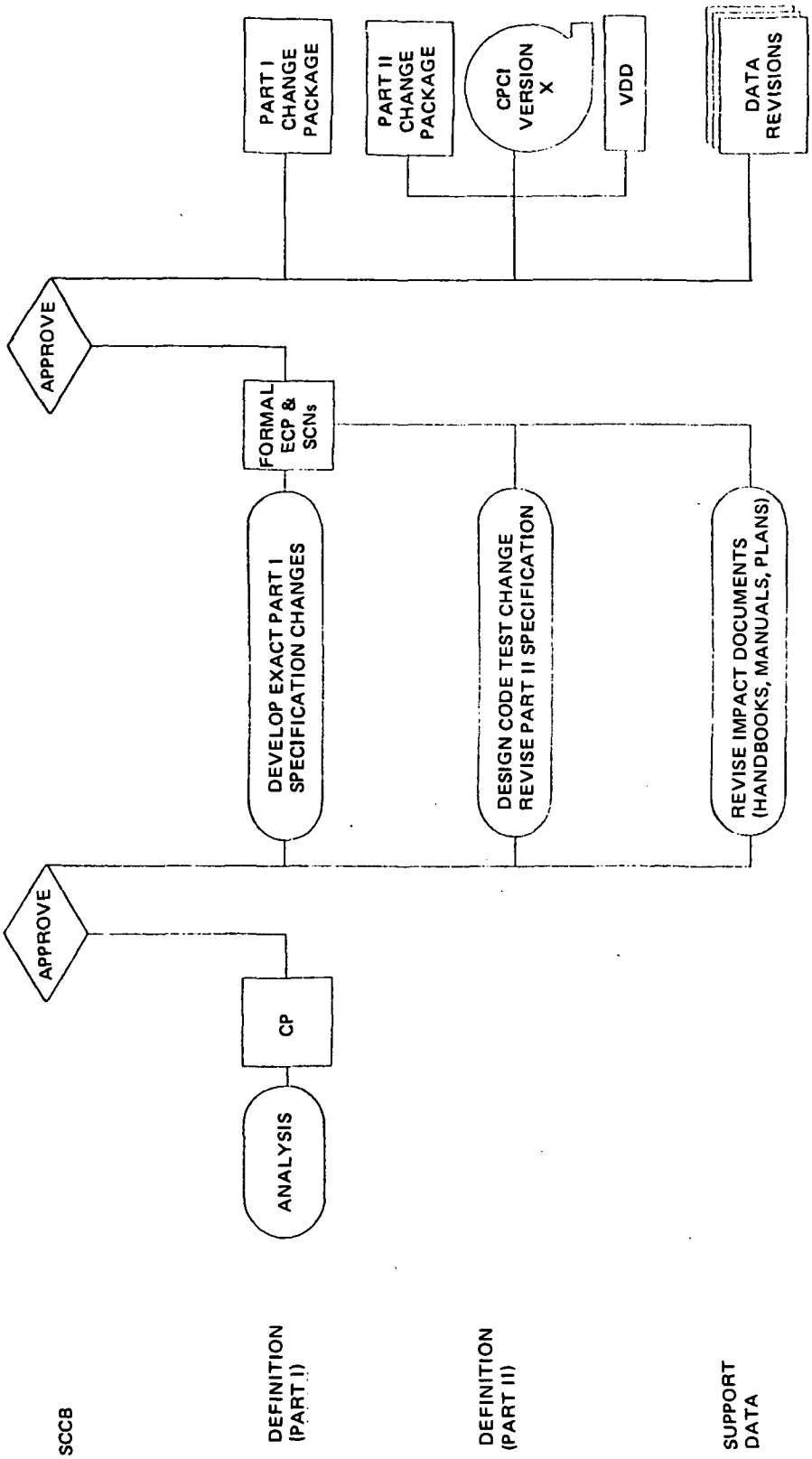


Figure 4-16. Computer Program Change Process

a. The sequence is shown as it might occur for a combination change, after the product baseline is established and support data items have been issued. Prior to that time, the Part I specification change would follow the sequence shown at the horizontal level labeled "Definition", but the change would be reflected in other items later, in their initial or basic issues.

b. A proposed change to a higher level specification, if involved, would be submitted with the initial CP, together with the appropriate SCNs.

c. Class IIA changes affecting the Part II specification only may occur. In these cases the Part I specification effort shown would naturally be eliminated.

d. The box labeled "SCNs" actually represents all final products of the change, as listed at the far right in the diagram. In the normal case, these would be completed and reviewed for approval over some distributed period of time, rather than simultaneously.

4.10.6 SPECIFICATION AND STATUS REPORTING

Based on the nature of the MSS program, and experience, it is anticipated that changes to computer program baselines will occur regularly and in considerable volume throughout the design phase, controls are designed to insure that the specifications and all important associated software documents will be maintained to reflect all changes. As indicated earlier, additional controls will insure that actual computer program configurations used in the system are known at all times, in relation to their approved specifications.

Each participating organization will issue, maintain, and distribute documents, and issue periodic status reports, relating to its area of responsibility for CPCI's, throughout the program. Detailed instructions will be prepared to govern uniform practice with respect to the areas summarized below.

a. Software Specification Maintenance

Document updating will be accomplished either by (1) incorporation of change pages or (2) complete reissue in the form of revisions. The normal method will be by issuance of change pages, except when a complete revision is specifically directed by the controlling authority. Instructions will cover: preparation of specification change pages and revisions; numbering; change package designators; preparation of SCN's; approvals and distribution; relations to change proposals and reports. Proposed and approved SCNs will be employed to cover changes to CPCI and other specifications, following the format and content requirements similar to those set forth in MIL-STD-490. SCNs will require approval prior to distribution.

b. Version Description Document (VDD)

The responsible engineering activity will be required to prepare a Version Description Document to accompany each new version of a CPCI or delivered set of changes. The VDD will encompass: description of the delivered item contents; identification of incorporated changes; installation instructions; description of possible problems or known errors; and other data relevant to status and use of the item. All VDD's issued will be reported in a Configuration Index document. A file of VDD's will be maintained centrally to provide a record of the actual computer program configurations which have been approved for use in the system. Instructions will cover format, content, timing, and distribution.

c. Configuration Index

The Configuration Index will be a document issued periodically containing a current listing of basic issues and updates of specifications and selected impact documents, together with identification of all changes incorporated in each issue. For each document, it will also contain a section which monitors impending future updates to reflect approved changes not yet incorporated. Instructions to be prepared will specify format, coverage, timing, distribution, and preparation, including applicable automated procedures for the Index maintenance and printing.

d. Change Status Report

The Change Status Report will provide a periodic summary of the nature and status of all change proposals relating to CPCI's for which an organization is responsible. Issues of the Change Status Report and Configuration Index will be concurrent. Together, these two documents will provide comprehensive software status information to all MSS activities, throughout the course of the program. Instructions will specify timing, organization of sections, and distribution of the report, including applicable automation of the status data.

5.0 RESOURCE ALLOCATION AND UTILIZATION RECOMMENDATIONS

5.0 RESOURCE ALLOCATION AND UTILIZATION RECOMMENDATIONS (WBS 94012-4)

Approaches for effective utilization of all MSS related data processing facilities are examined. Consideration was given to on-board data reduction/evaluation, and/or shuttle transfer-to-ground for ground-based data reduction, analysis, and distribution for both MSS vehicle operations and on-board experiment operations. Computer-assisted methods and procedures for (1) allocating resources such as crew skills, consumables, sensors, etc., within identifiable constraints, (2) scheduling resupply and crew rotation, and (3) long-range planning such as mission management and short range planning such as crew duty schedules are described.

This task consisted of the following steps:

Analyze MSS Resources and Scheduling Requirements - The MSS program was reviewed to determine the various facilities and resources which will require allocation and scheduling. These resources will include those required for MSS vehicle operations and for on-board experiments such as crew skills, consumables, and sensors. Consideration was given to the reduction and evaluation of data onboard and on the ground after transfer by the Space Shuttle.

Analyze Resource Allocation, Scheduling, and Planning Techniques - Documentation describing existing resource allocation, scheduling and planning schemes was collected and analyzed. A concise description, giving objectives, advantages, disadvantages and method, was prepared for each scheme.

Recommend Computer-Assisted Methods - Computer-assisted methods and procedures most suitable for the allocation, scheduling, and planning of MSS resources were recommended. This was accomplished through modeling decision-making tasks. This activity was based upon the information compiled in steps one and two.

Special MSS System Factors - Certain factors inherent to the MSS operational concept that will drive the software requirements were identified for future evaluation and development.

The objectives of this task were to identify and study computer assisted approaches for effective utilization of all MSS related data processing facilities and to recommend those approaches, methods and procedures most suitable for implementation and/or further development.

MSS system allocation, scheduling and planning requirements are presented in Section 5.2. These requirements include those necessary to perform mission and experiment operations both onboard the Space Station and all supporting ground-based facilities. These requirements are presented under the three subheadings of: (1) Users, (2) Processed Data and (3) Resources.

Section 5.3 presents a condensed but comprehensive discussion of past and current computer-assisted allocation, scheduling and planning technology. Past technology is discussed primarily in respect to problem areas and experience with implemented systems. Current technology is discussed in respect to the approaches being utilized and the basic operations performed.

Recommended data processing techniques and procedures for the planning scheduling and allocation aspects of an operational Modular Space Station (MSS) total system are presented as conceptual and capability requirements in Section 5.4. Specific examples of recommended concepts and capabilities, in respect to MSS requirements, are discussed and illustrated with the recommendations. Following the recommendations, an implementation summary is presented in Section 5.5.

5.1 ASSUMPTIONS

Modualr Space Station (MSS) requirements and the preliminary system design configuration as documented in the series of technical reports produced by NR for the Phase B Extension--Modular Space Station Program Definition--were assumed for this report.

The recommended computer-assisted allocation, scheduling and planning system assumes an initial operational capability (IOC) for the MSS. Pre-launch, launch and pre-IOC activities were not considered in defining MSS allocation, scheduling and planning requirements.

The Ground Network and synchronous satellite communications system assumed for MSS support is that defined by NASA for planning purposes in a memorandum dated May 21, 1971, entitled "Model for Ground Network and Synchronous Satellite Communications System," by D. R. Lord.

5.2 MSS DATA PROCESSING ALLOCATION, SCHEDULING AND PLANNING REQUIREMENTS

All individuals, or groups of individuals, who control the acquisition of, manipulate or apply operational and/or experiment raw or processed MSS data are considered to be "data users". Principal MSS data users will be:

- Space station crewmen who will monitor, control and manipulate data for station operations and experiments
- Ground personnel who perform command and control functions between the ground support network and the space station
- Ground personnel who perform central processing and data distribution for MSS operations and experiments
- Scientists and engineers who develop and modify operational and experiment procedures and interpret and apply the generated data
- Agencies and individuals who will apply the data to practical problems

The data generated and processed by the operational MSS system has been divided into two major types. These are: (1) Mission Operations Data and (2) Experiment Data. Operationally, the same generated and processed data may be required by the users of each major data type.

5.2.1 Data Categories

Principal users of mission operational data will be the onboard crew and ground mission management personnel. Primary data-user requirements for normal conditions are for real-time and near-real-time data to indicate the operational condition of the total system and its ability to perform current and future activities. For non-normal conditions, real-time and near-real-time data is required to assess the extent of degraded capability, its potential danger, and laternative methods for correcting the conditions which degrade mission objectives. Non-real-time data requirements will be for analysis and evaluation of previously performed operations and/or to determine future performance, positions and capabilities.

Mission operations data-users will require the major portion of their data as CRT displays or hard-copy printouts, with an ability to request major modifications in these outputs to meet special needs.

The number and type of experiments planned for the MSS are extensive and complex. Experiment data-user requirements will depend heavily on the nature of the experiment, the time period in which it is to be performed and the analysis and evaluation of previously performed experiments.

An Experiment Data Facility for the control and evaluation of MSS experiments is planned to be colocated to the MSS Mission Operations Facility. Depending on on the specific experiment, data-users of experimental data will vary considerably. Experiments in which real-time and near-real-time data is required will have the same data user requirements as real-time mission operational data and users will consist of station crew members, ground operational control personnel and experiment control personnel. As the real-time requirements of an experiment are diminished, the data-user requirements for onboard and mission operation personnel should correspondingly diminish; they simply assure that the experiment data is being acquired.

The MSS software concept for experiment data processing has assumed that each experiment will be defined as an entity to interface with the space station and ground support computer programs. All data acquisition, processing and dissemination requirements of onboard and ground-based mission operations are to be defined for each experiment. The experiment control and processing center will disseminate experiment data to a user system of Government agencies, universities and industrial requestors. The center will serve as the coordinating point for all experiments to be performed on the MSS.

A wide variety of both raw and processed data will be required to meet MSS data user needs. For efficient data management, all data items should be described and defined as a structure of data packages. The basis used to structure data products in current large-scale complex systems is time-of-need and facility capability. Wherever possible, data packages should be

standarized in format in order to meet similar requirements of different users. An overview of time-dependent data items to meet MSS data-user requirement is presented below.

Real-time data processing pertains to the actual time in which the related activity or process transpires. The data is used primarily to monitor and/or guide the activity being performed. MSS mission operations data acquired and processed by the onboard Information Subsystem (ISS) and the ground support portion of the total MSS Information Management System (IMS) have been designed to operate primarily as a real-time system. Individual data items produced in the real-time mode are numerous and constitute a list too large for separate identification. However, all items can be considered as constituting a large operational environmental data base. This data base, generated and maintained in real-time contains all data items required for automated and manual space station operations. Some space station experiments will require real-time data processing for experiment command and control. These data items should be considered as part of the real-time environmental data base.

Time related data acquisition and data processing is an arbitrary classification for data items required within some specific time span but not real-time. In general, any data processing required from a few seconds to 48 hours after the event in which data for the event was acquired is considered to be a time-dependent data item. Processed data required beyond the arbitrary 48 hours is classified as off-line.

MSS data-users will require a considerable amount of time-dependent data items. Major areas of use will be:

- Performance assessment and analysis of a previous activity to determine future actions
- "Quick-Look" evaluations of activities and events, just completed, to determine degree of successful achievement and/or problem areas
- Fixed period status reports of total system condition and capabilities
- Reconstruction of historical conditions and performances to determine future capabilities
- Update of initial conditions for future activities and events

Depending on specific time-dependent data item requirements, primarily the dependent time span, this type of processing is usually accomplished at the primary data collection site if workload capacity and capability exist. When primary site computation cannot be achieved, the data is usually sent via a data link to a secondary support facility.

MSS mission support and experiment operations will require large amounts of time-related data processing. Much of this type of processing will be specified through operating plans and procedures and will be formally scheduled as a part of a specific operation. However, system operational experience has shown that the computational loading for this type of data processing is subject to large variations. This is particularly true as system operational experience is gained and new and/or modified data requirements are requested by data-users.

Off-line data processing and the data items it generates are generally considered to be non-time-dependent in respect to system operations. In past systems this type of processing was performed only by a processor independent of system operations. However, with very large capacity processors and the utilization of multiprocessing and time-sharing techniques, this condition is no longer true.

MSS operations management off-line data processing will include detailed operational analysis and evaluation of past performance and status data, future position determination, logistics and other support, long range planning and contingency simulations.

MSS experiment off-line data processing is anticipated to be a major area of activity. While some real-time and time-related experiment data processing will be performed, the major use of experimental data will be in off-line processing where detailed analysis and evaluation can be made. Ths MSS experiment control center will perform this off-line processing in order to provide data items in a format suitable for the ultimate experiment data-user.

5.2.2 Resources

The major resources that will require allocation and scheduling during Space Station missions are (a) data processing capabilities, (b) onboard consumables, (c) onboard and ground-based sensors, and (d) personnel skills. Discussions of utilization implications for these resources are presented in the following sections.

In discussing the data processing resources that must be employed for the Space Station program, it is important to note the distinction between the use of onboard computer-assisted methods to perform the scheduling function itself, as compared to the scheduled employment of these data processing resources to support station operations and experiments. Implications of these two areas are as follows:

a. Performance of the Scheduling Function

It is envisioned that the onboard scheduling functions will be performed on the Station Operations Central Processor of the Data Processing Assembly. There are two functions which involve resource allocation, both of which are non-critical Information Subsystem operations. Characteristics of these functions are as follows:

Function	Required Operating Memory	Required Mass Memory	Anticipated Operations
Planning and Scheduling	2K Words	21K Words	Background
Logistics Inventory Control	2K Words	8K Words	2.3 KEAPS

Of the 23K total words allocated to Planning and Scheduling, it was assumed that 18K words would be required for instructions, while the remaining 5K would be for fixed and variable data. The 10K Logistics Inventory Control function was broken out into 7K for instructions and 3K for data.

It should be noted that these onboard estimates were predicated on a large proportion of the scheduling function being performed on the ground. As discussed

> "... Sizings are based upon the assumption that long range and comprehensive weekly and monthly planning and scheduling activities will be performed on the ground with necessary data transmitted (data link or shuttle delivery) to the Space Station. The onboard planning and scheduling routines will provide for control, manipulation, and modification of the ground generated data. ..."

It is quite possible that a heavier load of onboard scheduling requirements than those indicated above may be imposed on the Space Station, especially if it can be shown that interactive onboard scheduling is an efficient means for controlling the utilization of Station resources. This would raise the storage and speed estimates and thus impose increases in the overall memory and CP speed requirements for the Central Processors. The tradeoffs and implications of heavier onboard scheduling requirements will be discussed in subsequent sections of this report.

b. Employment of Data Processing Resources for Operations and Experiments Support

The emphasis throughout the ADT effort has been on the analysis and sizing of the Station Operations Central Processor. Since Station operations have been comparatively well defined, specific estimates of the loads imposed on the DPA by the seven operational subsystems were compiled and used to size the speed requirements of the CP, as well as for sizing of operational, mass and archival memory requirements. The assumption has been that experiment support will be performed by the second CP (the Experiments Central Processor), but that the architecture and component parameters of this second CP will be identical to that of the Operations CP.

Thus significant initial configuration parameters for the "2x" configuration (twice the basic sizing estimates) for each CP are as follows:

Speed requirements	critical functions	434K EAPS
	non-critical functions	828K EAPS
Operating memory requirements	critical functions	72.6K Words
	non-critical functions	61.4K Words
Mass memory requirements	all functions	682K Words
Archive memory requirements	all functions	8.4M Words*

Since current ground rules call for the duplication of all critical Station operations functions on both CP's, it can be seen that the Experiments CP will be capable of performing 828 KEAPS for support of experiments during normal conditions. (After a failure in the DPA, it is anticipated that the remaining CP will perform all critical Station operations functions.) It is this capability of 828K operations per second that must be efficiently allocated for the performance of candidate experiments - note that since the basic sizing analysis of the CP was based on the accommodation of a simultaneous worst case load of station operations functions, there will be no need for allocation of the CP capabilities for these functions.

Thus, the primary area of interest in scheduling onboard or ground-based data processing facilities is centered on the capability of the processing elements to support the monitoring, control, recording, and evaluation of experiments data; current plans are to involve Space Station crewmen mainly with monitoring and control activities, rather than with extensive processing of experimental data. Although this may be a reasonable ground rule, there could be "moments of opportunity" where rapid evaluation of results are required, necessitating crew and CP interaction. In addition, extensive onboard preprocessing and compaction of data may often be necessary prior to transmission to the ground. In these types of cases, it may be necessary to judiciously schedule CP availability to meet mission requirements.

Although extensive analyses of potential Space Station experiments has been documented in SD 71-217-3, most studies have involved the user interface, input and output data requirements, power and logistics requirements, and the like. Comparatively little analysis has been performed to determine the adequacy of the available CP speed and memory for direct support of onboard

*The Autonetics tabulation for archival memory apparently did not include totals for the Experiments Support Package. According to Table 2-5 of the Autonetics Report, this figure should be 7.7M words for the "basic" configuration, or 15.4M words for the "2x" configuration to support experiments.

experiments. It thus appears that an effective scheduling and allocation system should be implemented in order to ensure that experimental functions may be performed within allowable time constraints. (It should be noted that in Table 7-1 of SD 71-217-2, it is stated that experiment operations requirements will require 1045K operations per second. This figure was not covered in more detail in that report, but it appears that even in a 2x configuration, judicious allocation of the available power of the Experiments CP will be required for experiment support.)

It appears that little attention need be directed towards the allocation of the various buses onboard the Space Station. It is anticipated that Station Operations will contribute about 400K bps to the digital data bus, while experiments may contribute a worst case load of as much as 5000K bps. Thus, the 10 megabit capability planned for the digital data bus will be more than ample, and no conflicts should occur. Similar adequate design margins are envisioned for the telemetry bus, the audio/video bus, and the paging/ entertainment bus.

Thus, the main area of concern will be the efficient scheduling and utilization of the portion of the Experiments CP that will be employed for experiments support (i.e., that portion that will not be performing redundant Station operations computations). The efficient use of this facility - its arithmetic and input/output units, plus its operating, mass, and archival memories - should be determined by effective scheduling techniques.

5.2.3 Consumables

As part of the Logistics and Inventory Control function, monitoring and scheduling of the use of onboard consumables will be required. This will ensure the efficient use of these resources throughout the intervals between Shuttle resupply missions. The major consumables, as indicated in Table 4-3 of SD 71-221, are as follows:

- Clothing
- Linens
- Grooming Needs
- Medical Supplies
- Utensils
- Food
- Gaseous Storage (Oxygen and Nitrogen)
- Experiment Supplies
- Water
- Special Life Support, LiOH
- Water Management Supplies
- Atmospheric Control Supplies
- CO_2 Management Supplies
- Waste Management Supplies
- Hygiene Supplies
- Spares

The same kinds of consumables will be employed for both the initial and growth versions of the Station. Of course, the inventory, scheduling and control problem will be more complex in the 12-man growth version.

It should be noted that an effective logistics control and scheduling scheme is strongly dependent on an accurate "perpetual inventory." That is, automatic data entries for consumed articles (e.g., electrical connectors or sensors that can transmit the statuses and quantities of consumable products)

should be employed, and should be augmented by manual key-in of entries for other products, such as experiment supplies and utensils.

Since the maintenance and scheduling of consumables is closely connected with the day-to-day activities of the Space Station, this kind of resource allocation would probably be best exercised by onboard personnel, rather than through ground support (especially since this function does not require extensive computing time or facilities).

5.2.4 Related Experiment Support

To perform Space Station experiments and to support Station operations, a wide variety of onboard sensors will be required. In general, sensors used for operations support are separate and distinct from those used for experiments; i.e., devices that would be used to assess and maintain the status of the vehicle (attitude sensors, accelerometers, etc.) would not be employed for the sensing of observational data for experiments. However, it should be noted that many experiments will have a requirement for the recording of pertinent vehicle operations data to provide auxiliary information for those experiments. As an example, Space Station orientation, as determined by sensors associated with the G&C subsystem, would be an essential type of data that would be required for the proper interpretation of earth observation data. Thus, the scheduling and resource allocation function must consider the availability of "experiments" sensing devices. Discussions concerning each of these areas are as follows:

a. Operational Sensors

Although proposed Space Station experiments may require a wide variety of related operational data from all seven subsystems, the majority of sensors that could be used to provide auxiliary data for experiments are associated with the G&C subsystem (augmented by data from the RCS subsystem) and the ETC/LSS subsystem.

The purpose of the G&C subsystem is to provide the following functions:

1. Determination of the position and velocity of the station
2. Generation of guidance commands necessary to maintain the Station's orbit and to position free-flying experiment modules
3. Attitude pointing and rate stabilization (the G&C subsystem interfaces with the reaction jets of the Reaction Control Subsystem, which implements the G&C commands)

To perform these functions, the G&C subsystem utilizes data from several sensors in its various assemblies that can be used for a variety of experiments. These are:

. Accelerometer outputs (inertial reference assembly)
. Star tracker outputs (optical reference assembly)
. Horizon tracker outputs (optical reference assembly)
. Telescope/sextant outputs (optical reference assembly)
. CMG and gimbal parameters (control moment gyro assembly)

Numerous software commodities useful for experiments will be tabulated utilizing data from these sensors, including state vector parameters, inertial position and velocity, L.O.S. angles, experiments module position, and many others.

The other subsystem that may be used to produce auxiliary experiments data is the ETC/LSS subsystem. The primary function of the ETC/LSS is to provide a habitable environment for crew and equipment. Thus, where onboard environmental conditions may impact on the interpretation of results of certain experiments (e.g., contaminant measurement in animal modules, temperature and humidity in man-fatigue testing, etc.), recording of pertinent ETC/LSS sensor parameters will be required.

A variety of ETC/LSS parameters can be so samples, including the following:

. Temperature, humidity, and pressure controls
. Contamination measurements
. CO_2 removal and status

As with the G&C subsystem, ETC/LSS measurements and controls will be executed through the central processors and associated RACU's of the Data Processing Assembly.

b. Experiments Sensors

A broad range of sensor types will be available on the Space Station for experiments. As documented by TRW Systems as part of their Experiment Data Ground Processing Study, these sensors can be grouped into major wavelength regions:

1. Visible Spectrum

Film Units (framing, strip, and panoramic devices)

Electro-Optical Units (framing, electronic scanning and mechanical scanning units)

2. Infrared and Ultraviolet Spectrums

Point Detectors with Mechanical Scanning (imagers, radiometers, and spectrometers)

Detector Arrays with Electronic Scanning

3. Microwave Spectrum

Passive Multichannel Microwave Radiometer

Synthetic Aperture Side Looking Radar

4. X-Ray and Cosmic Ray Spectrums

X-ray telescopes, x-ray polarimeters, cosmic ray detectors, etc.

One of the areas in which Space Station sensors will receive extremely heavy usage is the area of earth surveys. As is documented in SD 71-217-3, there are at least 18 types of earth survey sensors envisioned for the initial station:

- Metric Camera
- Multispectral Camera
- Multispectral IR Scanner
- IR Interferometer/ Spectrometer
- IR Atmospheric Sounder
- IR Spectrometer/Radiometer
- MW Scanner Radiometer
- Multifrequency MW Radiometer
- MW Atmospheric Sounder
- Radar Imager
- Active/Passive MW Radiometer
- Visible Wavelength Polarimeter
- UHF Sferics Detector
- Absorption Spectrometer
- Laser Altimeter
- UV Image/Spectrometer
- Radar Altimeter Scatterometer
- Photo-Imaging Camera

These sensors can be employed for a wide range of experiments in the areas of agriculture/forestry, cartography, mineralogy, water resources, meterology, and oceanography. For example, as pointed out in Tables 7-4 through 7-9 of SD 71-217-3, the metric camera can be used for agriculture and forest inventories and disease damage in the agriculture/forestry area, for land use and urban planning in the cartography area, and has several other potential applications in other areas. Similar multi-use capabilities are available for most of the other sensors listed above. Thus, it is apparent that effective scheduling techniques must be employed to make optimum use of these experiment resources. It should be noted that as TRW has indicated in their study, increased requirements and attendant data rates should be expected as the Space Station evolves during its proposed lifetime. Although extensive study can be anticipated in this area, it is apparent that reasonable near-real-time scheduling and allocation methods must be found to achieve the ultimate in Space Station usability.

c. Related Schedulable Parameters for Experiments

Although sensor availability is a significant factor in the scheduling of experiments, there are several other multi-purpose commodities which complicate the efficient performance of onboard experiments. A basic list of the commodities to

be considered for computer-assisted scheduling of experiments has been listed in SD 71-217-3 as follows:

1. Electrical Power Requirements (average, sustained, and peak powers)

2. Crew Support Requirements (this will be covered in more detail in Section 2.3.4)

3. Output Data Rates (average and maximum)

4. Data Disposition Requirements

5. Data Input Requirements

6. Logistics Output Requirements

7. Logistics Input Requirements

8. Major Subsystem and Operational Requirements (e.g., G&C attitude control, ETC/LSS control)

All of these commodities will impact on the performance of experiments, and all constitute resources that must be properly allocated.

d. Ground-Based Sensors

The major category of sensors that will support Space Station activities will be the tracking and monitoring antennas and associated equipments of the supporting ground tracking network. The primary scheduling problem in this area will be to ensure that all required equipments are available to support Station passes (or relay satellite passes, if applicable) when these passes occur. Thus, checks must be made of available receiving antennas, telemetry receivers, subcarrier discriminators, converters, etc. to receive downlink tracking, communications, and telemetry data, as well as the required modulators, transmitters, command encoders, transmitting antennas, etc. to send uplink data.

This kind of sensor scheduling has received extensive analysis in the past. SDC has been involved in multi-station scheduling for the tracking net of the Air Force Satellite Control Facility, and has also had several years experience in the design and development of an interactive multi-station multi-satellite scheduling system for NASA's STADAN network. These techniques will be treated in more detail in subsequent sections of this report.

5.2.5 Personnel Skills

An extensive analysis of crew capabilities and operations has been documented in the MSS Phase B Extension documents. This report indicates that each of the initial six "positions" (commander, flight controller, systems engineer, electromechanical technician, electronic engineer, and experiment coordinator/scientist) will have a basic skill or background, but will be called upon for a multitude of tasks.

Twenty-seven basic skills are envisioned for the conduct of applications and experiment operations. The time phasing of experiments and the growth from a six-man to a twelve-man station results in an "average" number of skills per crewman at any one point in time of 1.9 during the initial station era, and 1.5 "average" skills in the growth station.

The twenty-seven basic skill categories that are envisioned for each version of the Station are as follows:

1. Biological Technician
2. Microbiological Technician
3. Biochemist
4. Physiologist
5. Astronomer/Astrophysicist
6. Physicist
7. Nuclear Physicist
8. Photo Technician
9. Thermodynamicist
10. Electronic Engineer
11. Behavioral Scientist
12. Chemical Technician
13. Metallurgist
14. Material Scientist
15. Mechanical Engineer
16. Electromechanical Technician
17. Medical Doctor
18. Optical Technician
19. Optical Scientist
20. Meteorologist
21. Microwave Specialist
22. Oceanographer
23. Physical Geologist
24. Photo Geologist
25. Physical Chemist
26. Agronomist
27. Geographer

It is evident that a reliable inventory and allocation scheme for available crew skills be implemented for allocation by (and for) onboard personnel. Such a scheme must, of course, consider current and anticipated experiment involvement of crew members with selected disciplines, along with availability as a function of allocated time for work, sleep, food preparation and eating, personal hygiene, recreation, exercise, and medical care.

The problem of scheduling ground support personnel is not nearly as complex as the onboard allocation function, since the ground support complex has a far wider range of personnel that can be called upon. However, it still will be necessary to schedule appropriate specialists for real-time Station support (i.e., during passes of the Station or relay satellite over ground stations), as well as for post-pass analysis and support. A partial list of the kinds of support individuals that may be required is as follows:

1. Tracking Data Analysts
2. Communications
3. Astrodynamics and Orbit Determination Specialists
4. Medical Doctor, Life Support System Specialists

In addition, several categories of experiment support personnel may be required for real-time communication and near-real-time analysis of downlink data. These categories would include the biologists, physiologists, chemists, astronomers, etc. that would parallel the onboard skill categories. It is envisioned that certain skill categories will be more time-critical than others; e.g., meterologists, oceanographers, and other earth survey personnel may very well be subject to stringent time constraints and may pose scheduling problems that necessitate computer-assisted support to determine personnel allocation.

5.3 COMPUTER-ASSISTED ALLOCATION, SCHEDULING AND PLANNING TECHNOLOGY

The problem of allocating resources to attain some predetermined objective has and continues to receive continued theoretical and applied study in the Operations Research field. Through theoretical studies of decision-making, optimization, queueing and information theory, many important problems of resource allocation, scheduling and planning have been identfied. This report section presents a brief discussion of previous efforts and problems in the use of digital computers and information processing techniques to solve allocation, scheduling and planning problems. The section also presents a detailed discussion of current computer-assisted methods being implemented as computerized planning, scheduling and allocation software systems for large-scale dynamic operations.

Initial use of digital techniques to aid in the solution of planning, scheduling and allocation problems were computer program routines used to conduct marginal analysis studies, based upon economic supply and demand theories. These studies were primarily static performance evaluations and/or predictions, which used past or a'priori data. Results were normally presented as scheduling and allocation plans to be implemented manually by management personnel. As the complexity of the systems to be managed increased, it became evident that the planning, scheduling and allocation functions being performed manually could not satisfy the conflicting demands and that the time and effort associated with the scheduling and allocation activity were becoming a pacing operational item.

Expansion of the traditional supply and demand analysis was initiated by applying queue and server analysis to determine the tradeoffs between waiting time and idleness. For meaningful results, it was determined that only through a dynamic analysis could realistic results be obtained. Dynamic digital simulations to obtain useful information for planners were designed and implemented to study the optimization of scheduling and allocation. The dynamic simulations were laboratory tools developed to deal with the growing need for broad, systems concepts in strategy. A few attempts to use the simulations to schedule and allocate resources in actual operating systems

failed because the presence of the human factors in the organizational performance were not considered, i.e., only the inanimate factors of production were considered.

Primarily from the requirements to apply the systems approach concept to all Air Force developments in 1958, computer-assisted planning research was accelerated. Air Force support system planners specified a need for determining the mix of personnel, equipment, facilities and management resources that would most likely maximize the mission objectives of systems to be developed. This need resulted in the widespread development of dynamic system simulations on digital computers to study interactions of configurations, support factors, component designs, management goals, malfunctions, information requriements, policies, etc. These simulations were used to perform "Management Games" for optimizing the planning of proposed systems.

An outgrowth of the computer technology applied to the simulations developed for Management Gaming was its direct application to scheduling the resources, events and activities of existing and proposed complex operational systems. The dynamic nature of the activities and allocation requirements for consumer product production, airline flight scheduling, building construction, classroom and teacher assignments and project management appeared to lend themselves to computer-assisted scheduling. These systems were termed dynamic because they were characterized by the fact that the resources, events and activities to be scheduled were frequently, if not constantly changing, and by the fact that the amount of data required to make intelligent management decisions could not be handled manually.

Since 1960 a major effort has been exerted, both in terms of funding and manhours, toward providing software to accomplish the tasks of scheduling operations, allocating resources and operations planning. The basis for this effort was that the savings in time and manpower that would result from providing computer assistance for the scheduling task would more than justify the cost of implementing such a computerized system. Other considerations were increased system reliability and increased capacity due to efficient operations management.

Initial software approaches were to provide an automated system capable of scheduling all necessary elements of the total operation. This approach included automatic resolution of conflicting demands for resources with minimal operator intervention. The justification of these automated systems were founded on principles which, although theoretically sound, were not consistent with operational facts, the most important of which was effective response to changing (and often unpredictable) needs. Some of the deficiencies were:

a. All indirectly related support activities were treated as secondary to any primary activities and could be scheduled after the latter were firmly established. (This was not realistic since activities caused by equipment down time cannot be cancelled in favor of any higher priority job.)

b. Support facility and equipment preparation (set-up) time was considered a constant for most combinations. (The fact is that preparation time is a function of many factors such as current load, facility configuration, last support effort, emergency support, and type of facility)

c. It was assumed that conflicts could be resolved automatically using an elaborate priority scheme and a statistical approach without regard to specific operational requirements. (In reality, priorities changed constantly, sometimes by the hours, and statistically based decisions were often unsound.)

d. The software program was of such magnitude and required such a large amount of input data for each run that it had no real-time capability.

As a result of these deficiencies, the approaches to a fully automated system were reexamined.

The new approach to providing computer-assisted operations system management concentrated on the computerizing of the laborious manual operations, in order to provide operators and planners with immediate access to the information they needed for decision-making. This was achieved through software designed for the maintenance of a real-time data base. The data base included information on all operational aspects of the system such as: facility and equipment status and condition, resources required and used, secondary and primary activities and events in process and planned, etc. This approach, when first implemented, would not automatically make decisions regarding conflicts, except in certain specific areas, but would identify conflicts for operator decision.

By 1967 refinements in the system previously discussed coupled with advanced and faster computing and peripheral devices resulted in significant improvements. By the development of algorithms for specific scheduling requirements, the user was not able to actually "see into" the data base and alter it to determine the effects of proposed actions. Thus, the new computer-assisted operational management systems had the capability to alter the environment through which all activities were scheduled and to identify each activity by numerous status indicators, start/stop times, min/max and duration, user/function codes, equipment requirements, supplementary text, control requirements, cyclic request logic, and the like. The type of system just described is the basic type of computer-assisted scheduling system implemented in the late 1960's. Some examples of such systems include: network scheduling for satellite control; activities scheduling for manned space flight; resource allocation in management and process control systems; component and element scheduling of computers and instrumentation in range operations; scheduling events and activities in air defense operations; and scheduling strategic operations in a war environment. Each of these applications is characterized by a massive data base with many conflicting requirements or competing demands for the same resources.

Advancements to the computer-assisted scheduling systems described are currently being developed and implemented. A significant development in this area has been interactive scheduling for dynamic systems. This refers to scheduling in which automated scheduling algorithms cannot be mathematically defined and the judgment of individuals must enter into the scheduling process on a continual basis. The person(s) performing the task can interact directly with their data, allocating resources and modifying the schedule as necessary up to the time of implementation. Dynamic systems are characterized by the fact that the resources, events, and activities to be scheduled are frequently, if not constantly, changing. The interactive process lends itself to the use of a time-sharing computer system in which the scheduling process is operated from one or more terminals of a multi-terminal system.

As the operational system complexity increases, it becomes increasingly difficult to predict all of the factors which may influence priorities or adequately define all of the constraints. The interactive approach provides for the display of a schedule such that the system operators could accept or reject the computer generated schedule, modify that schedule (particularly in response to changes in requirements), or make priority changes. All of this is done on a cathode-ray tube (or other output) displays with a light-pen providing direct input to the data base and schedule information. A function or display selector is usually available for calling up any display from a display repertoire. Having requested a schedule display, the operator can adjust the schedule using the light-pen until he finds the most suitable set of conditions. While present interactive system concepts emphasize operator resolution of conflicts, the concept allows for computerized conflict resolution.

From the preceding section, some of the advantages of the interactive approach are readily apparent. In actual applications, many of the systems implemented have unique features specifically designed for their particular application. However, the system features and basic operations of the approach, for all implemented designs, are as follows:

a. System Features

- Generally applicable to any situation where events, resources, or activities need to be scheduled, particularly where there are overlapping or conflicting demands for resources.

- System is interactive in that the user has immediate cathode-ray tube display responses to his queries and can input information via the light-pen or typewriter keyboard. For example, most systems provide the ability to instantly call up additional structured or hierarchical data about items in the display such as specific equipments for which there is a conflicting demand. The operator is able to determine the effect of any action on the overall schedule before executing that action. An automatic error message further enhances this capability.

- System is user-oriented in that the displays are presented in terms of user's needs and the language is in the user's vocabulary. Additionally, any errors made by the operator are immediately displayed with appropriate visual and audible alarms, with a cursor on the CRT indicating the item in error. Both inputs and outputs are simple and are designed to reflect the operating environment of the user.

- Intrinsic to most system designs is the accommodation of constraints and priorities. For example, in a satellite network scheduling task, many of the activities such as the station-vehicle contact time are constrained by the trajectory of satellites and such activities may be assigned different levels of priorities. Scheduling takes into account both the constraints and priorities, and these are reflected in the displays.

- As a result of constraints and priorities, as well as the need to conduct many operations in parallel, conflicts will result. The ability to predict and resolve such conflicts is part of the basic design.

b. Basic Operations

The basic operations of any computer-assisted interactive scheduling system are best indicated in terms of the input/output data flow. Figure 5-1 is intended to represent a typical system. For this system the following basic operations would be performed:

- Data Base Formulation - construction of a data base containing a definition of resources, environmental data and pertinent system status information

- Data Base Retrieval - provide access to the information in the data base through various media, such as: CRT displays, printed listings, list tapes, telemetry and other communication formats.

- Data Base Control - provide control of the information in the data base via: (1) alphanumeric keyboards, (2) interactive light-pens on CRT displays, (3) magnetic tapes and (4) data cards.

- Data Base Maintenance and Protection - provide the data base management functions of:

  (1) deletion, addition, or modification of tasks

  (2) deletion of a series of tasks

  (3) slide the times associated with tasks

  (4) edit the data base

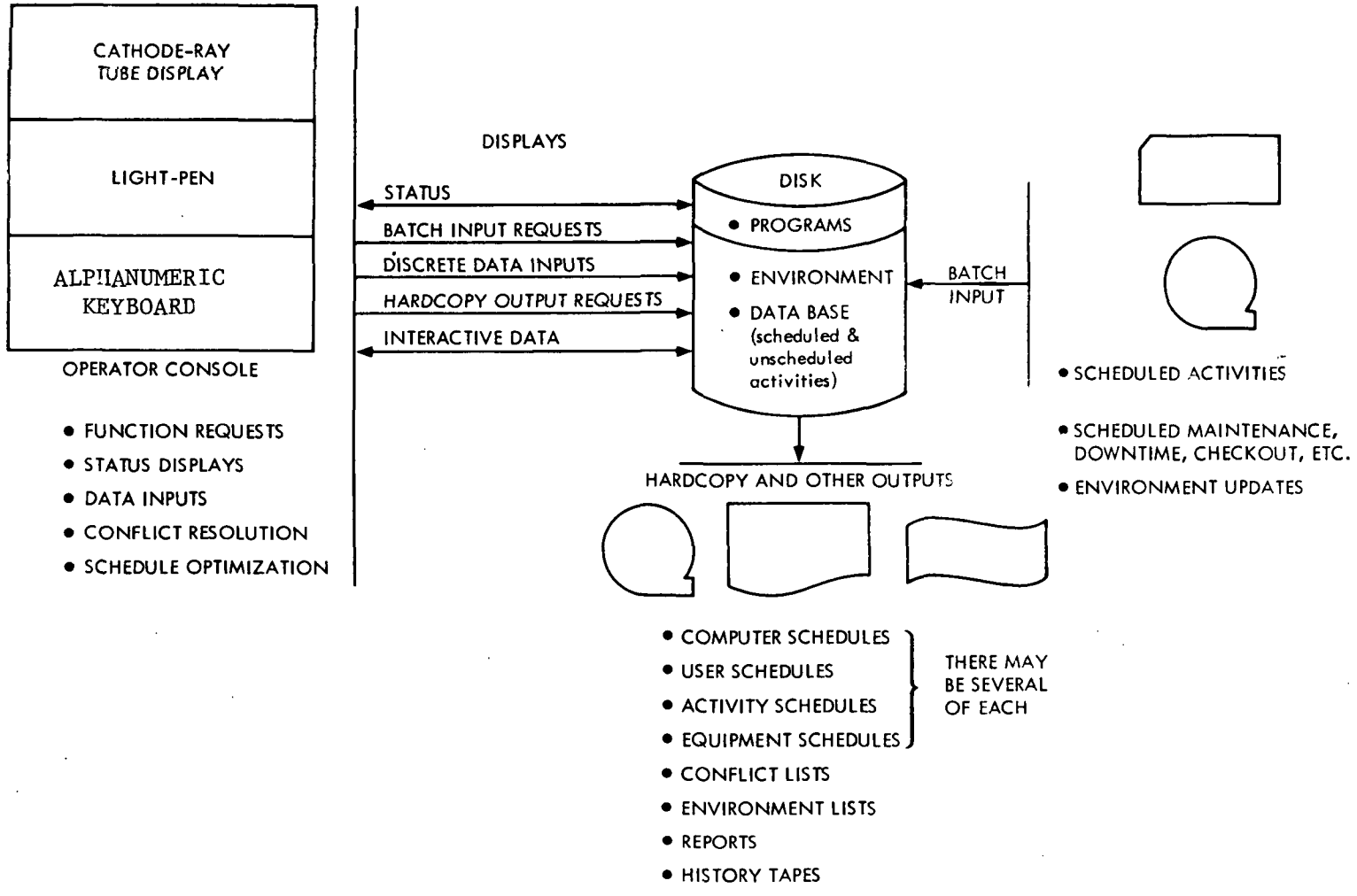


Figure 5-1. Input/Output Data Flow

- Scheduling - assign the resources of the system on the basis of available equipment and tasks already scheduled. Identify conflicts and reschedule on the basis of operator and/or computer directives.

- Conflict Resolution - identify activity, event and resource conflicts. Provide assistance in conflict resolution through conflict summary printouts and discrete modes of conflict analysis.

- Priority and Criteria Enforcement - prohibit operations, automatic or operator, which violage established priorities and operational criteria defined in the data base. Allow for priority and criteria change only under specific controlled procedures.

- Display Control - provide that the displays available through Data Base Retrieval can be assigned to any channel for subsequent call-up by monitors.

5.4 RECOMMENDED MSS COMPUTER-ASSISTED ALLOCATION, SCHEDULING AND PLANNING SYSTEM

Conceptually, the MSS system will differ from previous space projects in that major portions of the orbiting stations mission management functions will be performed on-board the station. The ground-based mission control and experiment control centers will monitor, back up and supplement the on-board mission management, plus serve as points of coordination of system users. Recommendations for computer-assisted allocation, scheduling and planning techniques presented in this section are based on the assumptions that the above described operating philosophy will be implemented.

5.4.1 Basic System Concept

To meet the planning, scheduling and allocation requirements of the total MSS operational system, a computer program system which provides a centralized source of scheduling information which may be changed or querried on demand is recommended. Significant features of this scheduling system should be:

- General applicability
- User-oriented
- Accommodation of constraints and priorities
- Conflict identification and resolution
- Generation and maintenance of current data bases

The total scheduling system should consist of compatible installations on-board the space station and at the ground-based mission control center. Input/output terminals will be located at each installation and consist of interactive display consoles equipped with functional keys, alphanumeric keyboards, light-pens, and channel selection keys to provide access to and control of the data base.

The data base should consist of the following basic categories of data:

- Environment Data - Current condition information on all system parameters, configuration, running time, etc.
- Task Data - Information on requested scheduled activities and required resources.
- Event Data - Information on past and predicted future events; performance, execution times, resources utilized and relationships to other events.
- Display Data - Preformatted displays which can be requested by the system operators.

On-line access to the large data base is the basic operating concept of the system. As new data becomes available, or as changes in scheduling requirements and resource conditions occur, on-line access to the data base provides console operators on-board the station and at mission control the most effective means of facilitating their decision-making responsibilities while at the same time increasing the span of control over the resource allocation and schedule implementation functions. Interaction with the operational data base is an integral part of this process. The system outputs "working" displays of operator selected data, providing computer generated responses as the interactive console operator performs additions, deletions, and modifications to the data base being presented.

Major capabilities of the system will be as follows:

a. Data Base Formulation - provides on-line input of system environment, scheduling requirements, selection and scheduling criteria, non-space station associated support requests, and batch input of space station information and post-event reports.

- Data Retrieval - provides access to the data base through various media.

  1. TV displays - space station characteristics, ground-station capabilities, space station activity plot, space station schedules, ground station schedules, priority lists, and parameter entry displays for data control functions.

  2. Printed listings - space station schedules, ground station schedules, space station characteristics, ground-station capabilities, priority lists, and printouts depicting results from data base editing functions.

3. Teletype format paper tape - space station schedules for use by the Operations Control Center, supporting stations and other system users.

c. Data Base Control - provides control of the information in the data base by utilization of alphanumeric keyboards and light-pen device for interaction with TV displays, user-oriented data and control cards for batch input or requests or non-event associated activities, and magnetic tape and/or disc files for batch input of event summary data.

d. Scheduling - the scheduling capability provided by the system will allow a maximum of user control over the selection and scheduling of required support activities. There should be two basic modes of scheduling within the system: discrete and algorithmic.

The discrete mode is defined here as the human process of "selecting" or "scheduling" one identifiable activity.

Conversely, the algorithmic mode is defined as the computer program processing of selecting and/or scheduling one or more identifiable activities during one continuous operation.

The discrete mode provides the capability of retrieving information on a particular event or activity, querying the system as to conflicts or constraints concerning that activity, and assigning a desired status.

The algorithmic mode provides facilities for (1) scheduling an aggregate of activities which have previously been "requested" as a function of the discrete mode, and (2) scheduling of space station functions which are selected by the program on the basis of scheduling requirements defined in the system environment. The first method searches the data base for all activities satisfying operator selected conditions (e.g., time span, ground-station, space station, support type, etc.) and whose schedule status is "requested." For each "requested" activity encountered, the function will change its status to "scheduled" if no resource conflict exists. If a conflict exists, a conflict summary printout will be generated. Conflict resolution is performed as a function of the discrete mode of scheduling.

The second method searches the data base for space station activities which satisfy the scheduling requirements defined in the system environment, and which are not in conflict with activities already scheduled, and assigns a status of "scheduled" to each selected task. In addition, this method schedules "requested" tasks where possible. When requirements cannot be satisfied, a conflict summary printout is generated.

Again, subsequent conflict resolution may be performed as a function of the discrete mode.

e. General File Editing - provides the console operator with general data base maintenance functions:

1. Delete, add, or modify individual support activities
2. Delete a series of support activities
3. Slide the support times associated with a series of space station passes forward or backward in time
4. Purge the data base, deleting support activities prior to a specified time

f. Display Control - provides the capability to assign any display appearing on an interactive console to any channel on the TV video drum for subsequent call-up by monitor positions.

g. Data Base Protection - certain data base protection features are provided to reduce the possibility of both interactive console operators attempting to modify support activities that cover the same time span. Requests for certain functions in the systems will be honored only when another console is quiescent, effectively "locking out" the other console for the duration of that function. Examples of functions falling under this restriction are purges of the data base, input on new pass summary information, slides of activities, and input of station post-pass reports. Generally this restriction will apply to any function which would effect massive data file restructuring.

Additionally, alarms will be displayed to the interactive console operator when he attempts to modify support activities whose start times are later than the beginning of the "next" daily schedule to be generated.

The same basic system capabilities should be provided for the onboard and ground-based portions of the total system. However, capability and capacity should be implemented to meet the following operational philosophy:

1. On-board space station planning, scheduling and allocation functions should be confined to a specific operational time span. This time span should be of sufficient duration to meet all but long-range planning needs.

2. Ground-based planning, scheduling and allocation functions should consist of support and verification of those performed on the space station plus long-range planning with optimization study and analysis, i.e., the ground-based scheduling system should function

operationally to support the space station, but should also be used to perform long-range studies of new and/or modified system requirements.

5.4.2 System Capabilities

The overall scheduling process will be accomplished both on-board the space station and at the Operations Control Center (OCC). Advanced scheduling will be performed by the advanced planning personnel while on the ground while those involved directly with mission control will deal with the implementation of the schedule and near-real-time scheduling. The scheduling process consists of two basic areas: "scheduling" related tasks, and "reporting" related tasks. Computer-assisted scheduling will have the capability of performing the following:

a. Review inputs to determine support requirements

b. Review special requests for specific operational support requirements

c. Predict requirements upon the data processing facilities

d. Coordinate all project requirements to develop a schedule of events for each facility

e. Ensure that all required predictions for support of schedule are sent to each facility

f. Maintain operational history of each event supported

g. Maintain historical files of all operations

h. Microfilm and store files as required

i. Format, publish, and distribute data to required users

j. Answer specific inquiries regarding acquired data

k. Produce various operational reports

Scheduling instructions will be maintained for each activity services by the system. These instructions set forth the manner in which an activity will be scheduled, both in the advance planning, and in any changes of the schedule after initial release of the schedule (i.e., rescheduling). Tasks can be scheduled or removed from a "scheduled" status at any time, using interactive displays.

Rescheduling will consist of two types, normal and real-time. Real-time rescheduling will be that scheduling performed within "T" hours of the impending vent. All rescheduling performed outside of this time should be classified as normal rescheduling.

Normal rescheduling usually results from a review of previously unavailable information and a resultant request for new coverage. Also, if previously scheduled activities must be cancelled, considerable bookkeeping is required since other operations may be dependent on the scheduled commands. Real-time rescheduling results from new requirements, malfunctioning systems, or support

station outages. Time is an important consideration in the real-time rescheduling effort.

Resource allocation and reallocation is an integral part of the scheduling and rescheduling process. Information on the availability and condition of all required resources is available from the environmental data base. Past history on their use is available in the event history data base. Through the interactive system any actual or proposed schedule can be displayed and the consequences of changes in allocated resources determined. Having requested a schedule display, the operator will adjust this schedule until he determines the most suitable set of conditions.

Reallocation like rescheduling results from a change in requirement and/or conditions for a previously scheduled activity.

Both operator and computerized procedures for conflict resolution should be provided by the system. However, operator resolution should be the principal method of solving schedule and resource allocation conflicts for the space station operational time period. Computerized resolution should be reserved for those cases where operator resolution cannot be achieved and for evaluating optimum solutions to long-range planning conflicts. Figure 5-2 presents a typical sequence of events for operator conflict resolution.

The recommended computer-assisted scheduling system will be used off-line to perform both long-range and short-range mission planning. Short-range should be considered as the planning for the next space station operational time period while long-range will be for future time periods. The planning function will be accomplished primarily by simulating future conditions with the scheduling system data bases and evaluating the effects of proposed future changes and requirements. To augment the planning function optimization software techniques capable of a higher degree of analysis should be incorporated with the scheduling system. Long-range planning should be primarily concerned with the following:

- Studies of vulnerability, reliability and operational capability for proposed system changes
- Analysis of competing configurations in respect to performance criteria
- Perform both optimal and near-optimal studies of scheduling

5.4.3 Input/Output Data Flow/Formats

The major elements of the recommended interactive scheduling system are the environment, task, activity, and display data bases. The basic content of these data bases have been incorporated in the baseline MSS system design to meet operational requirements. For planning, scheduling and resource allocation through the interactive system it appears that little new input data would have to be generated. However, existing input data may require additional structuring and organization. Input data aspects of the computer-assisted scheduling system are discussed in this subsection.

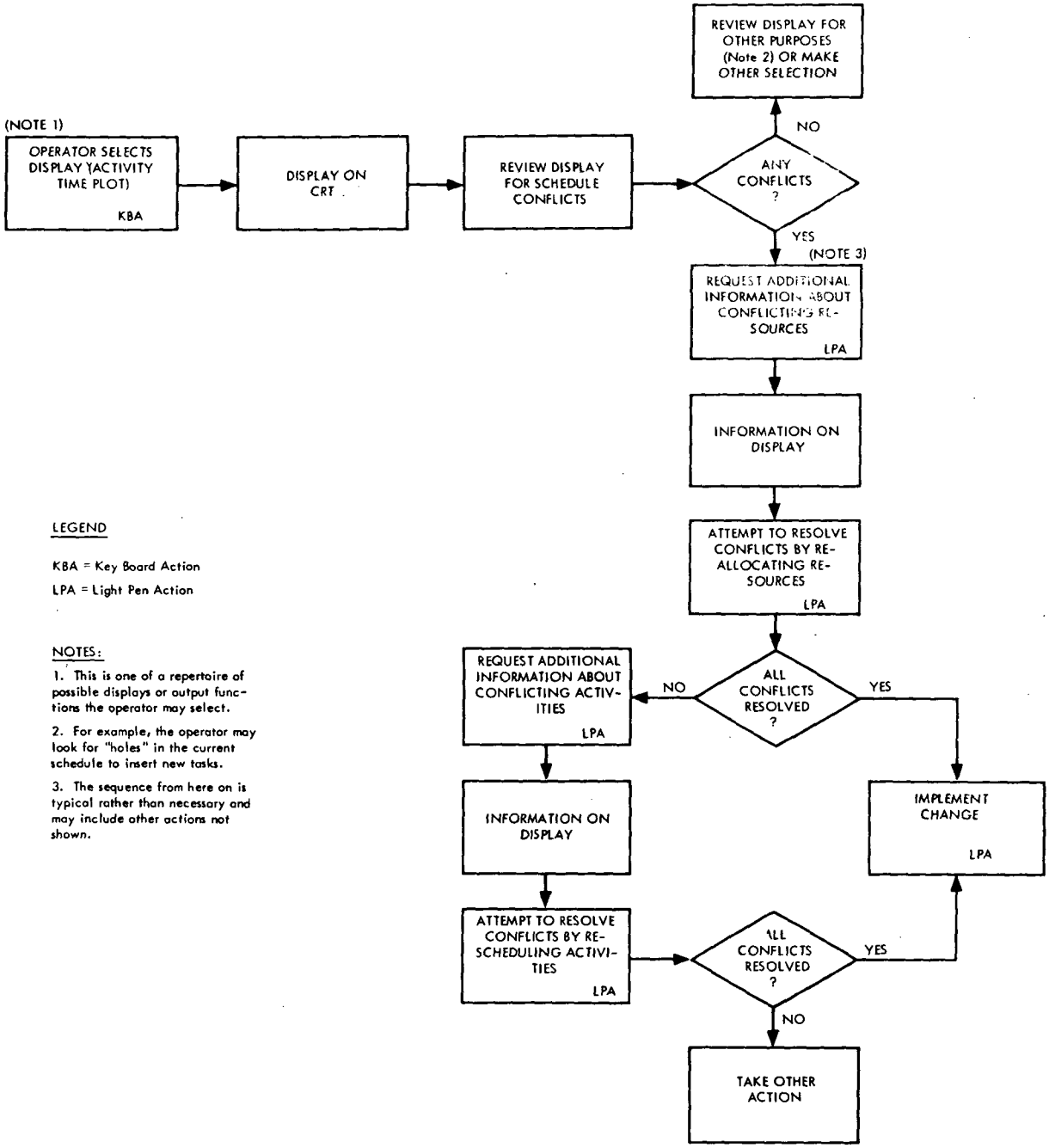


Figure 5-2. Typical Sequence of Events for Operator Conflict Resolution

Operation of the system consists of entry of information into a data base, display and modification of that information, periodic batch output of sets of information representing the current state of the data base, and eventual removal of information which is no longer required. The information entered into the data base from external sources will include prediction information describing activities and events as well as certain ground rules governing the scheduling processes to be used. Modification of information on displays will also be used to enter information from external sources. The displays will provide ready access to selected sets of information already in the data base. Modifications will, in the majority of cases, be made to effect requests for scheduling service. The principal item of batch output will be the schedule. The schedule can appear in a variety of forms; i.e., a Weekly Schedule Forecast to provide a basis for anticipating workload and service to be provided over a period of time; a Daily Schedule to show activity anticipated for a 24-hour period in the immediate future; Update Schedule Messages to show late changes made after publication of a daily schedule for a particular period.

Certain operations will normally be performed in a logical sequence through a weekly scheduling cycle; others will have no particular place in a cycle and will occur when required. Some data modification operations may occur at any time but should be done at particular times in the cycle. Bulk data entry, modifications of the environment data, and automatic scheduling itself should be periodically scheduled. A certain limited set of objects in the external world is relevant to the scheduling problem with which the system will deal.

The information necessary to catalog elements (station and spacecraft equipment items) with which the system is concerned will be contained in the environment data base. This environment data base will also contain certain rules governing the employment of these elements. A complete description of elements to be dealt with must exist before data entries describing events or tasks are constructed. As the system is put in operation, the current state of the system environment must be developed by means of interactive displays generated by the system.

At various times, the catalog of items receiving service will change, and the configurations of equipment at stations will change. The rules governing servicing will also change at times. Such changes in the relevant elements of the external world must be reflected by changes in the environment data base as they occur.

Advance planning usage will require that a change of equipment or conditions is associated with the time when the change is expected to occur. If an installation of equipment is anticipated prior to the next schedule period, the environment data must contain an indication of that equipment before it can be scheduled for use. When an equipment element is added to the data base prior to its installation, a corresponding "down" task must be entered for that task to preclude its scheduling prior to the time when it is expected to be operational. The same kind of "before-and-after" consideration applies when removal of an element of equipment is anticipated. When this occurs, a "down" task will be scheduled, commencing at the time of expected removal. After the equipment has been removed from service, the "down" task may be terminated and the environment data base modified by removing the indication of that element.

Various displays will present the data base information for the inspection of the user. These displays will all be interactive displays and may be used to add, remove, or modify information contained in the data base, when messages are received indicating changes will occur or have occurred.

It is anticipated that special display formats will be constructed for the purpose of displaying space station characteristics, ground-station capabilities, station parameters, station readiness, and other information which might be required at operational console positions. Interactive display consoles will provide capability for access to the data bases and modification of information in the data base on demand. Two classes of displays will be principally involved in actions at the interactive consoles. The first class includes those displays which provide control of program action: the Function Control Display and various Parameter Entry Displays. The second class includes those displays which present selected sets of information from the data base and provide for modification of information contained in the system tables. Several displays provide tabular listings of all environment data; the space station operations plot and ground-station plot provide a graphic representation of requested and scheduled activities as they appear in the task tables.

Display output requirements can best be illustrated by a typical interactive display. Figure 5-3 presents a typical interaction schedule display proposed for the SDC CAIRS* system. The display illustrated is a STATION/ACTIVITY TIME PLOT for a Satellite Control System. Of importance for the MSS system are the types of information presented as explained on the left side of the figure.

Specific displays for the MSS can only be defined through additional operational analysis. However, as an indication of the ability of an interactive computer-assisted scheduling system to provide for this output requirement, the SDC CAIRS system design provided for up to two hundred different displays.

Printer, paper tape, magnetic tape and other types of outputs will be required to meet specific system needs. It is anticipated that the majority of these type of outputs will be provided to meet ground-based requirements. Specific MSS hardcopy output requirements from the computer-assisted scheduling system will have to be determined and specified as operational requirements. However, it can be specified that they will be required to meet both system operational and data user needs.

5.4.4 Special MSS System Factors

There are certain unique MSS system features to be considered in the planning, scheduling and allocation process. A major section of the environmental data base will contain information for the management of space station consumables. Consumables required by the Environmental and Life Support Subsystem are in most cases continually monitored as critical operational functions. Their status will always be considered for any rescheduling of

---

*CAIRS: Computer-Assisted Interactive Resource Scheduling

EXPLANATION

Activity identifier column. (Names and/or numbers may be used.)

Indicates activities extends to preceding (<) or following (>) time period. Operator may request display for preceding or following time period.

Duration and features of activity. In the satellite tracking case, a line with two verticle markers (⊥—⊥) indicates the acquisition and fade times (the two verticle markers, respectively) of the satellite and the necessary station setup time preceding the first verticle. An open box indicates the vehicle is not scheduled for servicing whereas a closed box (⊥▭⊥) between the vehicle markers indicates that servicing is scheduled for that period.

A, B, etc., indicates configuration of resources to be used or are status codes.

Marker indicates activities selected for display in equipment or resources matrix (see below).

Control words (operator options associated with this display).

Marker indicates that operator has selected and is currently in EQUIPMENT mode.

Detailed information concerning first activity (N) selected by operator for equipment display. Operator may change situation by light-pen and/or typewriter action.

Coded equipment or resource list, read down, may extend to 48 items per display. In actual display abbreviations or short names of equipment are used.

Equipment assignment matrix. First two lines indicate if equipment is part of configuration A or B or shared (S). Next lines indicate equipment usage by activity (read down from coded equipment list above). Blank indicates equipment not at station, x indicates assigned, dot indicates unassigned. Operator may assign or unassign equipment by light-pen.

To input any change, operator would light-pen ENTER in control word line. To release equipment display or cancel inputs before they are entered, the operator light-pens anywhere along this line. All equipment markers and all items below the control word line would then disappear.

Alarm and status line. In this instance, operator has assigned equipment which is not at the station. There is an audible signal associated with the alarm.

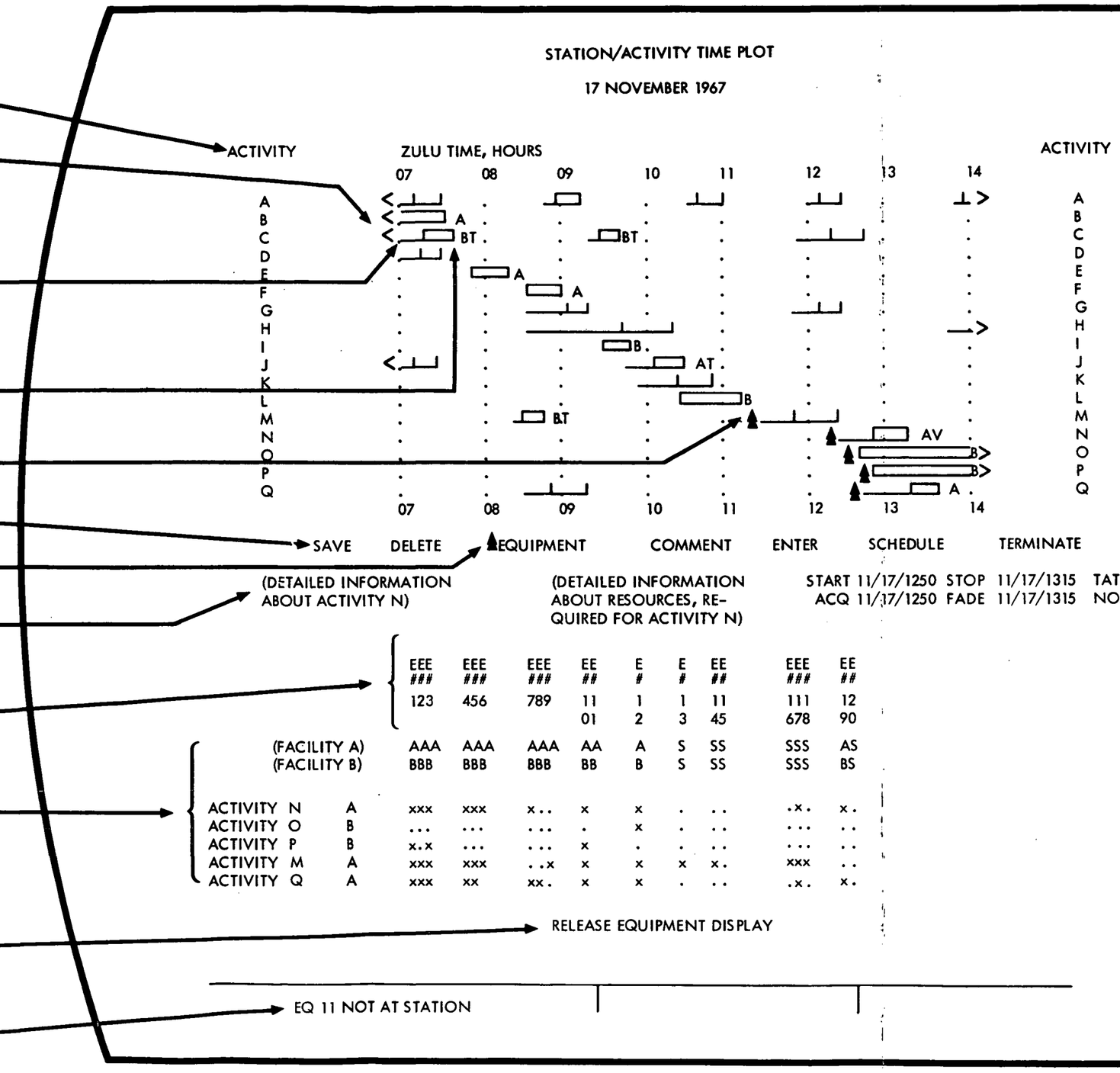


Figure 5-3. Typical Interactive Scheduling Display

activities because their consumption is directly related to the activities performed. The planning, scheduling and allocation of these types of consumables have been considered in the operational analysis of the space station.

In addition to the management of operational activity dependent consumables, those consumables used in the Crew and Habitability Subsystem which are subject to crew selection must be managed for resupply purposes. It is recommended that this type of consumables also be organized into a data base and that their status be available as a CRT display. Following the selection of an item from this type of consumable the crew member can use a light-pen to remove the item from the list of available items. This method can be used to maintain and keep records on the type of food, entertainment items and personal equipment used.

Total existing MSS system conditions will be available at any time to those performing long or short-range planning. These conditions can then be modified and changed in an incremental and iterative manner and the effects upon the total system observed by viewing various displays. For very long-range planning, the existing conditions can be modified to those defined for a future point in time and then performing an analysis of proposed changes.

As indicated in Section 5.2, various types of time-dependent data processing other than real-time processing, will be required to meet MSS system operational requirements. The computational facility to be used for specific time-dependent data processing products should be an integral part of resource scheduling. Within capacity and capability limits, all time-dependent data processing directly related to space station crew operations in the immediate future should be performed on-board the space station. When required, selected portions of these data products and/or crew analysis of the results can be transmitted to the ground. The objective to be achieved by this process is to limit and reduce the requirement to send data to the ground for processing where results must be sent back to the space station.

To achieve this operational philosophy, the status and workloads of the on-board data processing system for each activity must be included in the operational planning and scheduling process. Allocation criteria should be established to maximize usage of the on-board facilities and minimize space station to ground data transfer.

Off-line data processing as defined in Section 5.2 should be performed primarily at ground-based facilities. However, when real-time and time-dependent on-board data processing does not utilize total on-board data processing capacity, consideration should be given to performing this type of activity on-board the space station. This type of processing for operations and experiments, which are essentially independent of the time in which the results are required, could be scheduled as on-board processing. Results could be stored on magnetic tape and returned to earth via the resupplying Space Shuttle. The scheduling criteria for this type of activity should, as in the case of time-dependent processing, be the availability and capability of the on-board processing system and a requirement to limit the amount of data that must be transmitted via data links to the ground.

5.5 IMPLEMENTATION

A computer-assisted interactive planning, scheduling and resource allocation system of the magnitude recommended in Section 5.4 will require considerable development. Major portions of the required technology have been developed and implemented as indicated in Section 5.3. However, it is extremely doubtful that any existing system could totally meet the MSS system requirements even if extensively modified.

To provide a system of computer programs for planning, scheduling, and resource allocation functions of the total MSS system, it is suggested that the following major steps be performed:

Step A: Commencing from the date of MSS development initiation that a twelve month effort be devoted to a detailed study of MSS system planning, scheduling and allocation requirements, operational philosophy and criteria. Concurrent with this should be a review of current implemented systems for specific desirable features. Based upon these study efforts, a basic design document specifying criteria, capabilities and interface methods should be produced.

Step B: Following a project review and basic acceptance of the results of Step A, detailed design and development of the computer programs should be established as a part of the MSS computer program development plan. The computer programs required will be subject to the same development and testing procedures as all other MSS software and should be considered as an integral part of the total software system.

6.0 SUPERVISOR SPECIFICATION

6.0 SUPERVISOR SPECIFICATION

The purpose of this task was to develop preliminary performance characteristics of computer program modules needed to control the processors selected as part of the DPA configuration. Characteristics such as start-up, initialization, synchronization, "job" scheduling, input/output control, reconfigurability, monitoring and self-test, and shutdown/restart were considered for development. The objective is to develop an optimum supervisory (executive) design from both hardware and program consideration for the Modular Space Station (MSS) application. The data will then be documented in the form of a computer program specification for the DPA executive program.

6.1 DISCUSSION

The supervisory function is essentially that of scheduling and activating the various software tasks that perform all the system functions. This involves monitoring events, allocating resources and providing proper interfaces among separate software elements, among hardware elements and between the software and hardware. The supervisor must also respond to anomalous events, or errors, and initiate procedures to correct errors, isolate failures and, when possible, reconfigure the system to keep operating in spite of failures. In addition, the supervisor would maintain records for later analysis to identify hardware and software deficiencies that produce errors or otherwise adversely affect system performance. Since the supervisor itself is an "overhead" item that does not directly perform system functions, its intrusions (time and facilities used) should be kept to a minimum.

As digital computers developed in speed and capacity, it became apparent that one machine could process several jobs in a given time interval and this was more cost effective than using separate smaller machines. However, manual scheduling and control were inadequate, and so programs were developed to perform the schedule and control function. For ground-based purpose computers with many users and growing needs, these scheduling and control functions grew into massive operating systems that embodied many other functions and were generalized so that the same control and convenience software could operate in any installation using a given basic computer with any configuration of peripheral I/O and mass storage devices. In addition, they had to serve many different purposes for many users, even at a given installation. Such a general-purpose operating system may use half the time and capacity of any given data processing installation.

In parallel with the above was the development of dedicated, single application computers. The trend has been away from special-purpose, one-application computers to computers that are adaptable to a class of similar applications. Examples are process control, missile guidance, ship, aircraft and spacecraft navigation and guidance, weapon fire control, etc. For such applications, the software was (ideally) a one-time expense and multiple copies of it and the hardware would be used over a period of years. Therefore, the unit cost of hardware was more important than the software development cost and great emphasis was placed on reliability and quick response. For these systems, a general-purpose operating system was an unwarranted luxury.

As long as these applications could operate with a fixed, prescheduled sequence of programs, no supervisory program was needed. However, as the applications became more sophisticated and several more-or-less independent functions were performed in a given system, or the system operated in more than one mode, depending on external signals, and as the systems themselves became more complex, using active redundancy in hardware to achieve required reliability, more uncertainties were encountered. This required some active control to schedule different program modules at different rates and to rearrange the schedule and/or hardware/functional assignments in case of partial failures. Supervisory programs were developed to handle these problems and separate these considerations from the applications programs.

The Data Processing Assembly of the Modular Space Station contains elements of both dedicated special-purpose computers and general-purpose computers. Some of the Remote Processing Units will have fixed processing schedules and require no supervisor except, possibly, to handle communication through the Digital Data Bus (if this I/O is not strictly fixed periodic). The experiment processing, on the other hand, is as of now completely undefined and may require a fairly sophisticated supervisor.

These supervisory programs, in general, take care of uncertainties. In the initial design stage of the software for a given system, the supervisor will be supplemented by diagnostic and de-bugging programs to aid in tracking down hardware and software bugs. As the design progresses and experience is gained in use of and interactions between the various hardware and software elements and the timing involved, the requirements on the supervisor will be reduced. The complexity of the supervisor decreases as the uncertainties and ignorance of the system decrease.

One of the greatest difficulties in discussing software is terminology. New words are born or old words redefined with every paper written. Different organizations, or even different authors within the same organization, may use different words interchangeably or the same word with distinct meanings. They will use common English words, but nowhere explain what these words mean in the specific technical context they are used.

Supervisor - This is the subject of this report. It is, essentially, the central software control of the data processing system.

Executive - Often used interchangeably with supervisor. In this report, if used at all, will refer to a specific module in the supervisor.

Monitor - Sometimes used interchangeably with supervisor or executive. In this report it will refer to that portion of the supervisor that checks and maintains status information.

Operating System - Includes all the software that aids the user of a data processing system, and includes standards and conventions for programming, operating and information labeling. The supervisor would be part of an operating system which would also include standard subroutines, utility programs, language translators (assemblers and compilers), debugging aids, etc.

Bug - A mistake. Something inherently wrong in the design or implementation of either the hardware or software. A bug will always cause an error, although this error may not be evident except under particular circumstances. To "debug" is to remove the bug by correcting the mistake.

Error - An anomaly. Some state that should have not occurred has occurred. The symptom of a bug, a failure or a transient.

Failure - Something that was working properly but no longer does. Generally refers to some element of hardware that no longer performs its function within prescribed limits. An intermittent failure is one that appears and disappears (e.g., a loose wire making and breaking contact).

Transient - An apparent failure that does not recur. May or may not be detected as an error. Usually caused by some disturbance from outside the system and sometimes indistinguishable from an intermittent failure.

Higher Order Language - A means for expressing a procedure to be followed by a computer (a program) that is simpler (takes fewer coded instructions) than expressing the same procedure in primitive machine instructions.

Interpreter - A computer program that executes another program, usually expressed in a higher order language, by using a sequence of subroutines, one subroutine for each higher order language instruction.

Subroutine - A sequence of computer instructions that is used in many places in a computer program and/or by many programs. This common sequence of instructions is written and debugged once, thereby saving the effort of recoding the same, or similar group of instructions each time a given function is to be performed in a program. There are two general types of subroutines that used to be called "closed" subroutines (the adjective closed is no longer used) and "open" subroutines (now called Macros). Macros are common instruction sequences copies of which are inserted in the code wherever they are used. Subroutines (closed) have only a single copy which is executed by "calling" the subroutine. This "call" is a branch to the subroutine presenting it with information that tells it what data to work on and where in the program to return when done.

Call - See Subroutine.

Macro - See Subroutine.

Critical - Applied to programs and the MSS functions they support when necessary for crew safety and system integrity.

Public Data - Data always available to all software modules. Sometimes referred to as common or global data.

6.2 MODULAR SPACE STATION OPERATIONS CONTROL CENTRAL PROCESSOR SUPERVISORY COMPUTER PROGRAM SPECIFICATION (PRELIMINARY)

6.2.1 Introduction

This section describes the requirements of the supervisory programs in the form of specifications.

6.2.1.1 Purpose

The Supervisor is a set of program modules that monitor, service, and control the operation of a data processing system. Their aim is to keep the system usefully occupied as long as it is capable of operation.

6.2.1.2 Scope

This specification establishes the requirements for the supervisory software in the Data Processing Assembly of the Modular Space Station. In its present form, the specification is restricted to that part of the software that resides in the Central Processor dedicated to station operations.

6.2.1.3 Applicable Documents

The following documents form a part of this specification. In the event of conflict between these references and the contents of this specification, the detailed requirements in Section 6.2.2 shall be considered superseding requirements.

(a) DP 101 Data Processing Assembly (DPA) Configuration, 3-15-72
(b) AS 101 Modular Space Station Computer Standards and Conventions, 12-30-7
(c) AA 202 DPA Throughput and Authority Analysis, 2-11-72

6.2.2 Requirements

6.2.2.1 System Considerations

The Supervisor shall control the activity and assignment of all hardware and software elements within a central processor and shall monitor the status of all elements of the DPA. All applications software elements will be well tested and essentially debugged. The software will consist of application programs written to support MSS operations, the Supervisor program being described, and various utility programs and subroutines available to all programs. All programs shall be broken into independently executable modules each of which has a Module Control Block (MCB) that provides the supervisor with information needed to respond to and control each module. All internal application data will be either public or private, the public data being available in essentially fixed locations, and the private data for use within a module or for communicating between modules being in blocks temporarily assigned to the given modules by the Supervisor. Data for communication with elements of the DPA outside a central processor shall be kept in buffers set up by the Supervisor and maintained by the I/O processor programs.

Since the programs will all be cooperative and,essentially, debugged, it is not certain how much protection will be necessary for the information in memory. A certain amount of protection will be necessary for maintaining the integrity of critical programs and data, and checks may be necessary for some reading and many writing operations between M1 and M2 memories to prevent errors from propagating. The Supervisor shall be involved in such security, but the division of these functions between hardware (to save time) and software (for flexibility) is yet to be determined.

The Module Control Blocks shall have at least the following information for the Supervisor:

(a) Code name of the program module. Indication of whether it is subroutine.
(b) Activation indicator - a set of n bits to allow the conditional scheduling of this module by other modules or I/O operations.
(c) The number and identity of fixed-size data blocks required by this module for variable storage and inter-module communication.
(d) Which data blocks may be released on completion of this module.
(e) Code names of the program modules, each with a corresponding activation bit, which are to be successors of this module. If this module is a subroutine, the name (return module) will be provided by the module scheduling this subroutine. If a code name (e) is that of a subroutine, another name must be provided for the subroutine's return.
(f) A delay time - If the module to be scheduled is not time dependent, this delay may be zero, implying it can be scheduled without having to wait for the passage of time.

The Input/Output section of the central processor will be executing I/O programs in parallel with the Arithmetic Units. It shall communicate with the Supervisor by means of I/O Control Blocks (IOCB) which shall contain at least the following information:

(a) Identification of the source (Input) or destination (Output) of the information being transmitted and the route (DDB).
(b) The location and size of the buffer in M2 containing the information.
(c) An indication of the last word communicated (either a counter or a pointer to the buffer).
(d) Indications of any errors in the transmission, the numbers and sources of the errors.
(e) The name of the program module to be scheduled on completion of the I/O operation and activity and completion indicators.

The IOCBs shall be initialized by the Supervisor to start an I/O operation. The I/O processor maintains them from then until the data transmission is completed or the Supervisor terminates the operation. Certain IOCBs will be permanently scheduled to request inputs from elements of the MSS outside the DPA. Any error indications detected by the I/O processor shall initiate a recovery attempt which, if unsuccessful, shall treat the error as a failure.

6.2.2.2 Performance Requirements

This section establishes the functions that must be performed by the Supervisor. These are divided into the four general areas as shown in Figure 6-1.

6.2.2.2.1 Status Monitor. The Supervisor must be aware of the current status of all DPA elements. A monitor program shall maintain status tables from which the resource allocation and scheduling programs can obtain the information necessary for their decisions. In addition, the monitor will alert the crew and/or other subsystems of the MSS of any significant status changes that will affect station operations.

6.2.2.2.1.1 Passive Monitor. This portion of the monitor shall be called whenever an error has been detected by an interrupt or a program check. It will have three entries corresponding to transients, bugs and failures.

6.2.2.2.1.1.1 An error that disappears on retry will be classified as a transient. The Monitor will maintain time histories of such errors by source (hardware), location (software) and symptom (class of error indication). If the error rate exceeds some prescribed bound (may be different for each source, location, or symptom) the monitor will classify the error as a failure (hardware) or bug (software) and call the appropriate other entry.

6.2.2.2.1.1.2 An error that is consistent on retry but cannot be associated with a hardware failure will be treated as a program bug. The location and symptoms will be recorded and reported to the crew and the program module in which the bug was detected will be tagged.

6.2.2.2.1.1.3 A consistent error that can be identified as a hardware failure will be reported to the crew and the status entry corresponding to the failure will be updated. The required isolation and recovery routines will be called.

6.2.2.2.1.2 Active Monitor. This portion of the monitor shall be permanently scheduled on a periodic basis. It will perform checks on the hardware that is not automatically tested (e.g., comparators, parity checkers and other testing circuits) or has just been entered into service (either a new module or one just repaired). Requests for particular responses will be sent to each element of the DPA (RPUs, RACUs, and the alternate central processor set) and the lack of the required response within a prescribed time will be treated as an error. In particular, the operation and experiment processors must monitor each other so that the experiment processor can take over station operation if the other processor fails. Note that some of this system testing might be done with hardware.

6.2.2.2.1.3 Status Tables. Every In-Flight Replaceable Unit (IFRU) in the DPA shall have associated with it an entry that contains the following items to be maintained by the Monitor:

(a) Activity status (active or standby)
(b) Functional status (functional, failed, under repair test)
(c) Number of transient errors in a given period
(d) Allowable transient errors in a given period
(e) Time period for above (but only if different for different IFRUs)

Figure 6-1. General Supervisory Functions

If Item (c) is not zero at the end of the given period, it will be recorded, reported and reset to zero at this time.

6.2.2.2.1.4 Periodic Recording. This portion of the monitor shall be permanently scheduled. It shall record status summaries and certain critical variables necessary for the recovery from failures.

6.2.2.2.2 Request Response. The Supervisor shall respond to three kinds of requests for its services; direct requests from active program modules, indirect requests from I/O processor, and emergency requests through interrupts.

6.2.2.2.2.1 Direct Requests. The normal means for activation of the Supervisor will be through a direct call by one of the active applications program modules. In the multiprocessing environment, there are three possibilities, and the processor receiving the Supervisor call first must inhibit the other processor from also responding to a Supervisor call until the active Supervisor has determined the situation.

(a) The simplest, but least likely, situation is that the other processor has failed and is not operational. This can be determined from the Monitor's status table. In this case, nothing further need be done about inhibiting the Supervisor's response and the Supervisor proceeds in the single processor mode.

(b) If the Supervisor was called by a noncritical program module, the Supervisor response in the other processor shall be inhibited until released by the active Supervisor. Noncritical functions are processed in the multi-processor mode in which each of the two processors are executing independent modules. Since it is unlikely that the independent modules will be synchronized, and since the execution time for a typical module will be much greater than that of the Supervisor response, this inhibition should not cause an appreciable delay, if any.

(c) If the Supervisor were called by a critical program module, then both processors were executing the same module and should be synchronized except for any error responses that may be different in the two processors. In this case, the first Supervisor called will set a timer and wait for the other processor's Supervisor request. If this timer counts down before the second request, this event will be considered an indication of an error in the second processor. The second response shall stop the timer and initiate a comparison of the results of the two processors to verify that both are the same. All the variables transmitted to M2 by each critical module shall be checked in this manner.

Having determined the situation and performed the necessary checks, the Supervisor will remove the calling module's MCB from the waiting list, reset its activation indicators, and perform the memory management, I/O and scheduling functions specified in the Supervisor call.

6.2.2.2.2.2 Indirect Requests. When the Supervisor responds to the direct request, it shall also examine the IOCBs that are being updated by the I/O processor. Any I/O operations that have completed since the previous examination shall be deactivated or restarted if cyclic. The name of the module to be scheduled on completion will be passed to the Scheduler. Any error indications will be passed to the monitor.

6.2.2.2.2.3 Emergency Interrupts. Any errors detected within the DPA will cause one of the processors to be interrupted, the source and cause of the error will determine which processor to interrupt at what interrupt level. The Supervisor shall respond by saving all volatile status and register information, and then calling the monitor. In fact, the initial response to the interrupt may be part of the monitor. After the status tables are updated, the Supervisor shall attempt a recovery from the error, unless this recovery has already been accomplished by the hardware.

The recovery from a software bug requires active cooperation from the program in which the bug exists. If there are no alternate procedures, the entire program shall be isolated and the function it supports abandoned. All modules scheduled by this program that are on the waiting list shall be removed.

The recovery from a hardware failure is very dependent on what the failure was, what failures, if any, preceded it, and whether or not the program module in which the failure occurred supports a critical function. The details of the recovery from a hardware failure must be worked out carefully from software, hardware and functional considerations and cannot yet be specified here.

6.2.2.2.2.4 Memory Management. A program module releases a temporary data block when the information it contains is not needed by any of its successors. The memory space for this data block becomes available to other modules. The Supervisor shall keep a list of these available blocks, and add to this list any released by a module when it calls the Supervisor. Those data blocks not released shall be retained for use by successor modules.

In addition to the data blocks, it may be necessary for the Supervisor to load some program modules from M3 into M2. A decision will have to be made whether to trade space in M2 between data blocks and program modules, or to maintain fixed space for data blocks and to trade space only between program modules. This is discussed further under Resource Allocation.

6.2.2.2.2.5 Scheduling. The major scheduling decisions will be built into the application programs. Each program module will call for its successor(s) to be activated, and the MCB for each program module will contain an activation indicator specifying how many (and which) modules will have to call for it before it is activated. Each subroutine module scheduled through the Supervisor will have its successor (return) specified by the module that calls it. I/O operations will be treated as other program modules except their data blocks will be called buffers and their successors will be determined indirectly (see 6.2.2.2.2.2). Periodic modules will, in addition, have a time delay inherent in their MCBs or set by other modules.

The Scheduler portion of the Supervisor shall examine the request for successors of each program as it calls the Supervisor on completion, check that the request is legitimate (such a successor exists and a bit in its activation indicator matches that in the successor request), and set the successor's activation bit. In addition, the Scheduler shall check the IOCBs of the active I/O modules for similar successor requests. When all the activation bits for a given module are set, if its time (incremental delay or absolute clock time) is matched or exceeded by the Supervisor's clock, the MCB of that module shall be put on the waiting list in its priority queue to wait for the Resource Allocator to start its execution.

6.2.2.2.2.6 Timing. The Supervisor shall maintain an internal clock that will be periodically synchronized with an external clock (either part of the MSS or a time signal transmitted by a ground station). The resolution required for this clock is yet to be determined. In addition to the master clock, the Supervisor shall provide interval timers for the various time intervals required by the scheduler. These may be separate hardware countdown timers (noninterrupting) or timers maintained by the Supervisor by reference to a single countdown timer. The Supervisor shall also maintain countdown timers for the various testing functions required (e.g., program module timing, timing of responses from other DPS or MSS elements, etc.). The details of the implementation of these timers will depend on how many distinct countdown timers, interrupting or noninterrupting, are provided in the hardware.

6.2.2.2.3 Resource Allocation. The Supervisor shall maintain a list of available resources, including all elements of the DPA and time, and associate these resources with program modules to assure efficient use of these resources in the execution of the functional applications programs. The Monitor shall indicate whether a particular DPA element is operational. The Resource Allocator shall keep track of which program modules on the waiting list require what resources and shall assign currently unassigned resources in such a manner as to make most efficient use of these resources.

The resources to be allocated consist of: time, processors, and data blocks for program modules; I/O processor, DDB channels, RPUs, RACUs for I/O programs; and M2 or M3 memory space for programs to be loaded or data to be saved.

6.2.2.2.3.1 Reconfiguration. The monitor and recorder functions shall maintain proper status history records to support the resource allocation algorithms in both system degradation and system recovery. Whenever there is a change in the operational resources (an element fails or a repaired element added), the resource allocation algorithms must be aware of the event (through access to the monitor's status tables) and adapt accordingly. These algorithms will be sensitive to the sequence of events as well as the current status and must adapt in such a manner as to:

(a) Guarantee crew safety and MSS integrity
(b) Support critical MSS functions
(c) Guarantee proper operation of the DPA
(d) Support the experiment data processing

in descending order of priority. By using only those hardware and software elements that the status tables indicate are operational, the software portion of the reconfiguration will be accomplished. The physical switching-off of failed hardware elements shall be accomplished by both hardware and software in cooperation.

The details of software reconfiguration and, in particular, the adaption of the allocation algorithms to different sequences of bug and failure detection remain to be worked out.

6.2.2.2.3.2 Timing. The Supervisor shall start an interval timer each time a module starts execution, and stop this timer when the module completes execution. If a module does not finish execution within the specified interval, a timer error interrupt will occur.

Whether a single interval shall be used for all modules, or whether each module should have an associated execution time interval in its MCB is yet to be determined.

6.2.2.2.3.3 Processor. Since the Supervisor is occupying a processor while being executed, it will always have at least this resource to allocate. The other processor will not be available if either it has failed or it is executing a noncritical module. Allocating a processor is equivalent to starting the execution of a module. If both processors are operational, four situations can exist, depending on whether the module that called the Supervisor and the one about to be started are critical or noncritical. If both are critical, then both processors are available and both must be allocated to the critical module. If neither is critical, then only one processor is available, and it may be allocated to noncritical function. If the previous were critical and the next not, the Supervisor has two processors to allocate to two noncritical modules, and may do so. (Note that if critical and noncritical modules can have the same priority, and if the module after the next on the ready list is critical, there may be a choice of whether to hold the noncritical module until after the critical module has finished with both modules, or to make the critical module wait until the noncritical module is done so that two processors are available. If critical and noncritical modules cannot have the same priority, this problem cannot exist.) If the previous module were noncritical and the next is critical, then the critical module must wait for the completion of the noncritical module being executed on the other processor. In this case, if all modules are given the same time interval for execution, no problem exists. If variable time intervals are allowed, the allocation function may be complicated by having the allocator look for a waiting noncritical function that can be executed in the time left before the currently executing module is expected to finish.

6.2.2.2.3.4 Memory (data blocks). The Supervisor shall keep track of available data blocks. If any of the program modules in the waiting list require the allocation of data blocks, the available data blocks shall be distributed among them in the most efficient manner. All program modules on the waiting list are automatically ready for execution if they need no additional data blocks. Those program modules on the highest priority waiting list shall be allocated data blocks first, if there are enough available to make them ready. The allocation algorithm shall avoid the deadlock problem of partial allocation to several programs from a limited availability list.

6.2.2.2.3.5 I/O Processor. The Supervisor shall allocate the I/O processor to a particular I/O operation by activating the associated IOCB. The I/O processor shall maintain its own scheduling and timing by reference to the active IOCB list. Buffers will already have been allocated for I/O by the modules which scheduled the I/O; the data blocks allocated to those modules become the buffers for the I/O processor.

The I/O processor will take care of all I/O error detection and recovery.

6.2.2.2.3.6 Memory (Programs) and Mass Memory. As currently envisioned, there is no requirement for program loading and relocation. The M2 memory is planned to have the capacity for all program modules in its fully active status and for all critical program modules and data if one memory module fails. However, to allow for expansion, the Supervisor shall provide for the loading and relocation of program modules from M3 to M2. The replacement and overlay strategy shall be related to the scheduling and shall indicate in the MCBs the loaded status of the corresponding program modules.

The Mass Memory, M3, will be used for saving checkpoint data and status history. The Supervisor shall keep track of which locations in M3 have been allocated to which information. The oldest data will be overwritten by the latest, and selected portions shall be copied into archival memory, M4, for historical records and mission analysis.

6.2.2.2.4 Start, Stop and Error Recovery. The Supervisor shall provide for integrating all working elements of the DPA into a working system and to isolate all failed elements in a manner that will not adversely affect other subsystems of the MSS. This implies the ability to bring the system into operation from a cold start as well as accepting the replacement of a repaired IFRU that had previously failed, or a new IFRU to allow modular expansion of the system. The isolation of an element applied in the proper sequence to all elements allows safe shutdown of the entire system. These functions will require some assistance from the hardware and, in fact, the DPA is designed to take much of the burden from the Supervisor software.

6.2.2.2.4.1 Initial Start. The power-on signal shall cause an interrupt (or its equivalent) that starts the execution of a program at a specific memory location (implying that there is some independent procedure for initially loading the memory).

The initial startup shall then initiate some self-verification procedure for the processor receiving the interrupt followed by a verification of the memory modules and I/O processor. Status tables shall then be initialized and other elements of the DPA turned on and tested in a prescribed sequence to guarantee safe (no erroneous signals to other subsystems) buildup of the various functions. The program modules shall be checked and, if not properly loaded, read from either M3 or M4 and their MCBs and data block status tables initialized. The RPUs and RACUs will have their own power-on startup procedures and their status checked through the I/O processor. The internal clocks shall be synchronized with some external time signal. The monitor shall update the status tables and report and record the system buildup in its normal fashion. When sufficient system elements are operational, the scheduling and allocation portions of the supervisor shall initiate normal system operations.

6.2.2.2.4.2 Restart and Error Recovery. The restart and recovery procedures shall operate in a manner similar to initial startup except that the status tables and data are recalled from where they were saved rather than initialized and the element testing will be applied only to those elements (IFRUs) that have been returned to the system for replacement of a failed element. The difference between restart and recovery is essentially that restart implies the return of a processor to the system and recovery can mean either the addition or removal of a system element. In some cases the recovery is taken care of by the hardware and the only response required by the Supervisor is updating the status tables and allocation algorithms. In other cases the recovery will also require resetting data blocks and rescheduling program modules.

6.2.2.2.4.3 Stop. The Supervisor shall initiate power-off sequences for all failed elements that are not automatically turned off. The sequencing shall insure safe MSS operation and include the normal monitor reporting of the shutdowns. This implies that, for complete DPA shutdown, the last element turned off is the processor itself. The sequence shall allow, but not require manual intervention in the power-off sequence.

6.2.2.2.5 Data Base Requirements. The amount and format of the Supervisor's data base is not yet determined, but the following items shall be included.

6.2.2.2.5.1 Status Tables. A table showing the status of each element in the DPA shall be maintained. The information for each element shall include those items listed in Section 6.2.2.2.1.3.

6.2.2.2.5.2 Module Control Blocks. A table describing each software module in the program system shall be maintained. Each entry in this table shall consist of a module control block as defined in Section 6.2.2.1. Either copies of or pointers to MCBs shall be maintained in a Waiting List in priority queues for all modules scheduled and waiting for execution.

6.2.2.2.5.3 I/O Control Blocks. A table describing each I/O process shall be maintained. Each entry in this table shall consist of an I/O control block which will include those items listed in Section 6.2.2.1. Either copies of or pointers to IOCBs shall be maintained in an active I/O list for communication with the I/O processor.

6.2.2.2.5.4 Data Blocks and Buffers. The data blocks and buffers (which may be interchangeable, the name applied implying use by a program module or an I/O program) are properly part of the applications programs; but the Supervisor shall maintain a list of pointers to all data blocks/buffers with an indication for each whether or not it is allocated to a program module or I/O program.

6.2.2.3 Interface Requirements

There are two interfaces of concern to the Supervisor: with applications programs and with MSS subsystems (the display and control assembly represents the crew to the Supervisor).

6.2.2.3.1 Program Interfaces. All interfaces between applications programs and the Supervisor shall be program module level. Each module shall, on successful completion, call the Supervisor through its Direct Response. If the application detects an error during its execution, it shall call an application - specific diagnostic and recovery subroutine (or schedule this subroutine if it is a long one) which, in turn, will report its findings to the passive monitor before returning to either its calling module or the Supervisor.

The Supervisor shall schedule and activate each program module through control of its MCB.

6.2.2.3.2 MSS Interfaces. The interface between the Supervisor and the MSS subsystems shall be through I/O control blocks and their associated buffers. These, in turn, are updated by the I/O processor which interfaces the MSS subsystems through the DDBs and RACUs or RPUs. The Supervisor determines the status of each RACU or RPU by sending requests for responses and examining the responses. The MSS subsystems send requests for service through coded messages formatted by their RACUs or RPUs. These coded messages are stored by the I/O processor in the IOCBs and associated buffers. The Supervisor shall read these messages as requests to schedule particular program modules.

6.2.2.4 Design Considerations

The Supervisor contains modules (scheduling and allocation) that are activated frequently relative to other software and other modules that are (hopefully) rare exceptions (responses to errors and configuration changes). Every applications module activates the scheduling and allocation functions, which therefore requires that the interface be simple and their execution rapid to reduce the programming and debugging of the applications programs and to keep low the overhead time spent by the Supervisor. The error responses of the Supervisor must be done carefully and completely to enhance the fault tolerance of the DPA. For the latter, completeness is more important than simplicity or speed.

6.2.2.4.1 Programming Language. The Supervisor shall be coded in machine or assembly language. This will assure the maximum efficiency in the use of the data processor's features to attain speed of execution and completeness of error checking. The requirements for change in such a program are less extensive than those for applications programs, therefore a higher order language does not offer as much advantage. Also, many of the executive functions are highly machine dependent, which makes higher order languages, in general, unsuitable.

6.2.2.4.2 Programming Standards and Conventions. The standards and conventions as specified in Part 2.0 of this volume shall be applied. Good documentation must be stressed, since documentation historically has been the weakest point of most software developments.

6.2.2.4.3 Test Features. The Supervisor is unique in that it can include features to aid in testing not only itself but also all other program modules. Some of the error detection, status monitoring, data recording, and memory protection features included in the executive can be expanded to aid in program bug detection. These features can be useful in the development of the Supervisor itself, but will be even more useful in the development of the application programs. The debugging features of the Supervisor should therefore be developed early and designed in

such a manner as to be easily removable or reducible. Thus, they will not use up time and memory overhead as the system becomes operational and the debugging and testing features can be left on the ground.

6.2.2.4.4 Expandability. For the Supervisor, compressibility is more to the point. As the development proceeds and as experience is gained in operating the MSS, the requirements for the Supervisor generally decrease. On the other hand, the requirements for the Supervisor in the experiment data processor will include those in this document (since it may have to take over this function) and will probably be greater. Since the experiment processing requirements are unknown now and will probably change from experiment to experiment, the Supervisor complexity must take up the slack in the requirements uncertainties.

The Supervisor shall therefore be written in a modular fashion to allow both expansion and contraction as well as imbedding into a larger, more complex experiment Supervisor.